FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Voto seguro pela Internet através de *smartphones*

Pedro Sousa Grilo

Mestrado Integrado em Engenharia Informática e Computação

Orientador: José de Magalhães Cruz

5 de Março de 2015

Voto seguro pela Internet através de *smartphones*

Pedro Sousa Grilo

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Gabriel David
Arguente: André Zúquete
Orientador: José de Magalhães Cruz
5 de Março de 2015

Resumo

A tendência mundial de aumento do número de smartphones e acessos à Internet, providencia novas oportunidades ao voto pela Internet. Cada vez mais pessoas utilizam smartphones para realizar tarefas que requerem cuidado (e.g., o *Home Banking* e compras); atualmente, os sistemas de voto pela Internet começam a tornar-se mais seguros e viáveis. Aliando estes dois fatores, torna-se inevitável que sejam usados smartphones para realizar o voto.

Os sistemas de voto são sistemas críticos, exigindo alto nível de segurança. Por isso, qualquer implementação destes, requer cuidados especiais e que sejam cumpridos vários requisitos, que caso sejam violados, põem em causa a credibilidade das eleições.

Este trabalho está inserido num projeto desenvolvido pela Multicert, empresa de segurança informática e certificação digital, apelidado de CertVote, sistema de votação eletrónica através de navegadores da Internet.

O objetivo deste trabalho foi estudar e implementar um sistema de voto, que funcione com o CertVote e permita votar de modo seguro através de smartphones. O resultado final foi a criação de uma solução que inclui duas aplicações móveis e um servidor, alicerçado no CertVote. O sistema desenvolvido seguiu um conjunto de boas práticas e guias de testes de segurança. Durante o processo de desenvolvimento, foi implementado um módulo de assinaturas RSA num projeto *open-source* e foram descobertas e relatadas várias vulnerabilidades na aplicação Android de voto da Estónia.

São fornecidos ao longo deste trabalho, os detalhes da arquitetura e da implementação que foram utilizados, assim como os testes de segurança efetuados que permitem chegar à conclusão de que o sistema é seguro e está pronto a ser implementado em larga escala.

Abstract

The global trend towards increasing number of smartphones and access to the Internet, offers new opportunities for Internet voting. More and more people use smartphones to perform tasks that require caution (eg, the home banking and shopping); currently, the Internet voting systems begin to be more secure and viable. Combining these two factors, it is inevitable to use smartphones to vote.

Voting systems require a high level of security, as they are highly critical. Therefore, any implementation of these ones requires special care and must fulfill several requirements, which, if violated, question the credibility of the elections.

This work belongs to a project developed by Multicert, computer security and digital certification company, dubbed CertVote, electronic voting system through Internet browsers.

The objective of this work was to study and implement a voting system, that works with CertVote and allows to vote safely through smartphones. The end result was the creation of a system that includes a secure mobile application and a server that are fully integrated in CertVote. The developed system followed a set of best practices and security testing guides. During the process of development, an RSA signature module was implemented in an open-source project and were discovered and reported several vulnerabilities in Android application for Estonia voting.

Are provided throughout this paper, the details of the architecture and the implementation, which were used to unsure the safety of the system. The conclusion is that the system is safe and ready to be implemented on a large scale.

Agradecimentos

Gostaria de agradecer ao meu orientador, Dr. José de Magalhães Cruz, pela inesgotável disponibilidade, paciência e contributo no enriquecimento desta dissertação.

Agradecer à Multicert e colegas de trabalho, pelos recursos, ajuda e motivação.

À FEUP, por me ter proporcionado as condições necessárias para elaboração deste trabalho.

Ao meu querido irmão João pela ajuda incansável na revisão do texto.

À minha família, um enorme obrigado por acreditarem em mim e terem me apoiado e feito possível chegar aqui.

Ao meu pai e avô, que apesar de já não estarem comigo fisicamente, foram fundamentais para o meu rumo e para o que sou hoje.

"The ignorance of one voter in a democracy impairs the security of all."

John F. Kennedy

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

CBC	*Cipher-block chaining*
DDOS	*Distributed denial-of-service*
DEX	Formato executável de Dalvik
EBC	*Electronic codebook*
E-Voting	Sistema de voto eletrónico, com ou sem uso da Internet
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
IOS	Sistema Operativo da Apple para dispositivos móveis e Apple TV
I-Voting	Sistema de voto electrónico através Internet
JAR	*Java ARchive*
JSON	*JavaScript Object Notation*
JVM	*Java Virtual Machine*
MIX-NET	Canais Anónimos
PIN	*Personal Identification Number*
QR-Code	Código de barras em duas dimensões
SVE	Sistema de Voto eletrónico
SMS	Simple Message System
SSL	*Secure Sockets Layer*
TLS	*Transport Layer Security*

Capítulo 1

Introdução

No século 6 A.C., surgiu o sistema político da Democracia Ateniense com o ideal de criar "Um Governo do Povo, pelo Povo e para o Povo". Como estratégia para implementar o seu ideal, o conceito de voto foi introduzido como ferramenta primordial da Democracia. Nos dias modernos, o voto é cada vez mais utilizado como meio de decisão. O voto é usado por organismos públicos e privados e as decisões através dele tomadas afetam a vida de empresas, pessoas e países. Por ser um mecanismo tão preponderante em vários setores das sociedades atuais, é imperativo que o voto seja usado como meio de exprimir, apenas e só, a vontade dos eleitores. No entanto, desde o aparecimento das eleições por voz até ao voto pela Internet, várias técnicas e meios foram utilizados para manipular, alterar ou coagir o resultado de uma eleição. Por isso, é necessário que os sistemas de voto estejam preparados para ataques sendo assim a segurança, um ponto primordial no desenvolvimento deste tipo de sistemas.

O voto pela Internet, apelidado de I-Voting, já é amplamente utilizado pela sociedade atual. A Estónia permite aos seus cidadãos que votem pela Internet, através de um computador ou smartphone. A demanda por I-Voting está a aumentar e segundo o artigo "*The digital divide and Internet Voting Acceptance*", na Holanda, 62 porcento das pessoas preferiram votar pela Internet; o autor também refere que o I-Voting reduz custos, reduz a abstenção e o tempo de contagem dos votos.

Com o aparecimento dos *smartphones* é possível estar ligado à Internet em qualquer lugar, possibilitando assim o I-Voting em qualquer lugar. O cerne deste trabalho desenvolve-se sobre a utilização de *smartphones* como meio de voto pela Internet, sendo referidos os vários princípios criptográficos a utilizar, mecanismos de segurança e testes a aplicar ao software de modo a garantir a sua segurança.

1.1 Contexto/Enquadramento

Esta tese de mestrado foi proposta pela empresa Multicert. A Multicert é uma organização que oferece soluções de segurança e certificação digital. Os seus sectores de atividade vão desde o setor financeiro, até setores como o da administração pública e privados.

Os produtos oferecidos são: segurança de informação, vários tipos de certificados (eg., SSL e PK12), fatura eletrónica, CodeSigning e uma solução de voto eletrónico.

A solução de voto eletrónico é fornecida através de vários meios, como : o voto em ambiente Internet; mesas de voto eletrónico em ambiente presencial, com caderno eleitoral eletrónico; quiosque / cabine de Voto para votação eletrónica em ambiente controlado; e ainda, sistema de leitura ótica dos votos.

Em 2004, a Multicert foi pioneira em Portugal ao lançar um teste piloto de voto eletrónico para as eleições do Parlamento Europeu e, em 2005, faz também outro projeto piloto para as Eleições Legislativas. Ao longo dos anos, este sistema tem sido vendido a privados, estando desde então, em contínuo aperfeiçoamento. Como culminar do trabalho desenvolvido, chega no final do ano 2014, o CertVote (Figura 1.1).

Figura 1.1: Instantâneo de uma página do CertVote

Este sistema é uma solução de voto onde o ato de votar pode ocorrer exclusivamente pela Internet ou usando combinação dos vários outros métodos supracitados . A plataforma foi concebida seguindo várias normas de segurança. A solução é vendida em dois pacotes: o padrão, onde o cliente pode configurar e operar o voto de modo autónomo; e, o pacote *premium*, onde a solução é personalizada para moldar ao máximo aos requisitos dos clientes.

O aumento do número de utilizadores de *smartphones* e o aumento de ligações à Internet através deles, levou a Multicert a criar uma solução móvel que garantisse o voto seguro, confiável e anónimo.

A presente dissertação foi supervisionada pelo Eng. Ricardo Ferreira, da Multicert, que acompanhou o seu desenvolvimento e, academicamente acompanhada, na FEUP, pelo Prof. Dr. José de Magalhães Cruz.

1.2 Projeto

O projeto proposto foi divido em 4 artefactos. O primeiro é uma aplicação Android, que permite o voto pela Internet de forma segura. O segundo artefato é uma aplicação prototipo IOS que permite o voto. O terceiro artefato foi um servidor que faz de elo de comunicação entre os smartphones e o CertVote e o quarto e último foi a documentação interna e esta dissertação.

1.3 Motivação e Objetivos

Como já referido, a Multicert oferece aos seus clientes várias soluções de voto. No entanto, ainda não tem uma solução móvel para oferecer aos seus clientes. Como foi mostrado na introdução deste capítulo, o número de pessoas que desejam votar pela Internet está a aumentar, assim como a procura destes serviços por Governos e empresas. Devido ao aumento do número de utilizadores de *smartphones*, as soluções móveis de voto eletrónico vão ter um papel cada vez maior nas decisões por todo o mundo. Este protejo de dissertação foi realizado com o intuito de estudar os pontos fortes e fracos do voto em smartphones através da Internet e propor uma solução rápida, fácil de usar e de alta segurança para os eleitores.

Os objetivos propostos pela Multicert passam então por:

- Estudo comparativo de soluções existentes
- Desenho e conceção de uma solução
- Implementação de um prototipo

1.4 Planeamento

O planeamento do trabalho foi estipulado com a meta de dividir as principais tarefas por períodos de tempo. Na Figura 1.2 é apresentada a primeira parte do planeamento e na Figura 1.3 a segunda .

- Passo 1 - conhecer o CertVote (aprender e instalar as ferramentas necessárias para o correr). Durante este período, foi estudada a plataforma Spring para compreender como funcionam e como se desenvolvem servidores Java com ela e, ainda, foi estudado o uso do Maven, como gestor de dependências.
- Passo 2 - produção a parte do servidor responsável por verificar a existência de uma eleição e recolher os seus dados.

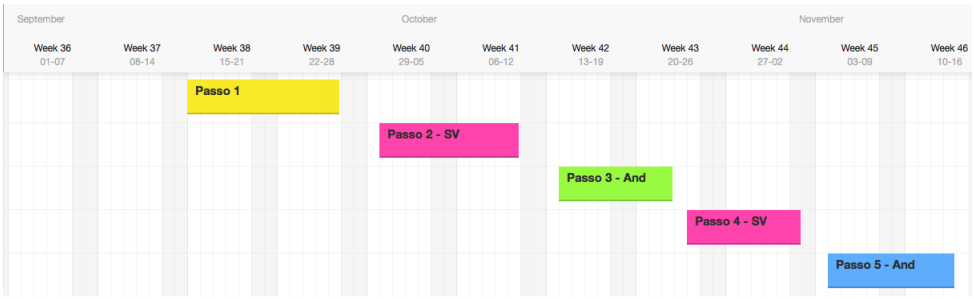


Figura 1.2: Planeamento - Parte 1

- Passo 3 - estudar o guia de boas práticas de desenvolvimento para Android, concebida a funcionalidade do QR-Reader e implementado o sistema de procura de eleição no Android.
- Passo 4 - elaborar funcionalidade de verificação de credenciais e a entrega dos boletins para o servidor foi elaborada.
- Passo 5 - criada a recolha das credenciais, geração de chaves e assinatura das credenciais.

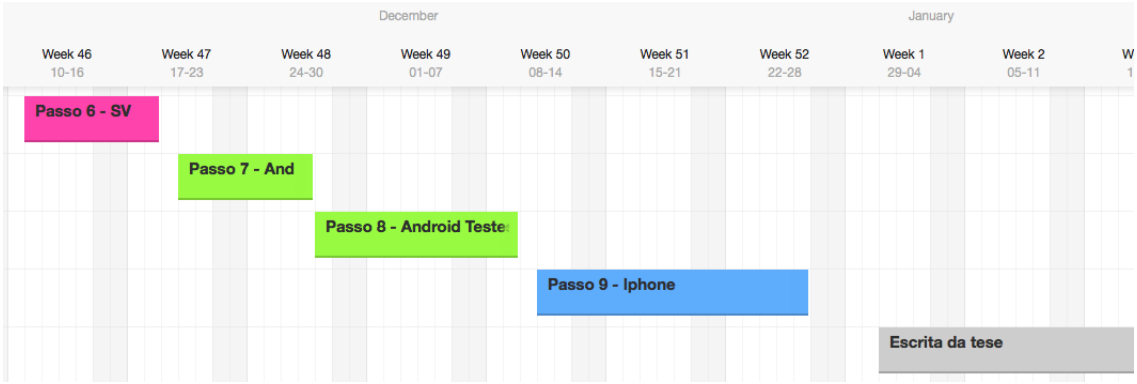


Figura 1.3: Planeamento - Parte 2

- Passo 6 - desenvolvido no servidor a verificação de tokens e envio do voto para o CertVote.
- Passo 7 - montada a recolha do voto, mostragem de opções escolhidas no boletim e implementado o uso de 3-DES.
- Passo 8 - realização de testes unitários, funcionais e de segurança à aplicação.
- Passo 9 - realizado o protótipo IOS.
- Passo 10 - dedicada à escrita deste documento.

1.5 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 5 capítulos e pode ser dividida em duas partes. A primeira parte está localizada no capítulo 2 e serve para representar o estado da

ciência nos vários aspetos ligados aos sistemas de voto, criptografia e sistemas operativos móveis. A segunda parte (restantes 4 capítulos), começa no capítulo 3 onde é exposto o problema que se pretende resolver, apresentado o sistema CertVote e definidos os requisitos e a arquitetura do sistema. Por último, é feita uma análise de segurança. No capítulo 4 são referidos os vários detalhes da implementação da solução, onde são descritas as tecnologias utilizadas, detalhes da implementação usados para cumprir os requisitos e tornar o sistema seguro. O capítulo 5 é onde são expostos os testes realizados para comprovar a segurança da aplicação e, para finalizar, o capítulo 6 é uma conclusão geral dos assuntos debatidos ao longo da tese.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é feita uma revisão bibliográfica de trabalhos anteriores sobre os temas abordados na Dissertação, são descritos os vários padrões já existentes e é feita uma análise comparativa entre eles, permitindo assim uma maior compreensão dos assuntos abordados nos capítulos seguintes.

2.1 Voto como problema de segurança

Ao longo da história, o sistema de voto tem sofrido várias mudanças com o objetivo de tornar as eleições mais seguras e evitar fraudes. Contudo, hoje em dia, com a implementação do I-Voting (voto pela Internet), existem mais ameaças do que nunca. Os sistemas de I-Voting têm de ser desenhados e arquitetados com o foco na segurança, pois são sistemas críticos. Eles têm de cumprir e garantir vários requisitos gerais, comuns as todas as eleições, e para isso, usam vários tipos de mecanismo de segurança. Os requisitos principais são os seguintes:

- **Elegibilidade** - Apenas eleitores regularizados podem votar.
- **Anonimato** - Não permitir descobrir quem votou em quem.
- **Integridade** - Após o voto ser realizado, ele não pode ser alterado nem eliminado.
- **Singularidade** - Cada eleitor tem direito a um, e um só, voto.
- **Não coercibilidade** - Nenhum eleitor pode provar em quem votou, para não existir compra do voto. Este ponto pode ser muito difícil de controlar no voto pela Internet.
- **Abstenção** - Qualquer eleitor pode decidir não votar .

Os sistemas de I-Voting têm mais requisitos que os tradicionais, pois têm necessidades e comportamentos diferentes como, por exemplo, necessitam de ter um mecanismo que permita auditorias externas. Em seguida, são enumerados vários desses requisitos [evB14]:

- **Auditabilidade** - Tem de existir um mecanismo, que permita verificar se o processo de voto e contagem correu de modo correto e, que possa ser verificado por entidades independentes.

- **Disponibilidade** - O sistema de voto têm de permitir a realização do voto durante um determinado intervalo de tempo, sem falhas.
- **Precisão e Robustez** - O sistema têm de calcular os votos de modo preciso e deve ser capaz de aguentar vários tipos de problemas, sem perder precisão.
- **Detetabilidade** - O sistema deve ser capaz de detetar falhas e intrusos. Isto permite, por exemplo, cancelar umas eleições caso exista o risco de votos alterados ou introduzidos na votação.

Os sistemas de voto foram evoluindo para colmatar as falhas dos anteriores, e assim, o estudo dos métodos de voto é uma etapa introdutória e bastante necessária para entender estes sistemas, porque permite compreender o comportamento e as técnicas usadas por quem os queira atacar. Com este conhecimento pretende-se que os novos sistemas sejam desenhados de modo a não se cometer as mesmas falhas. De seguida, são apresentados vários problemas encontrados ao longo da história, referenciados no livro "Broken Ballots: Will Your Vote Count?"[SB12].

2.1.1 Voz

Nos primeiros sistemas de votação, a voz era utilizada como meio para realizar o voto. O eleitor dizia em voz alta a sua opção de voto que era, depois, apontado pelos responsáveis, sendo mais tarde contabilizado .

Figura 2.1: George Caleb Bingham - The County Election

Problemas

Como é muito fácil compreender pelos parâmetros de qualidade dos sistemas de voto utilizados nos dias de hoje, este sistema de voto tem inúmeros problemas e o quadro da Figura 2.1 pintado por *George Caleb Bingham* retrata isso mesmo. O primeiro grande problema deste sistema é o facto de o eleitor ter de expor o voto, permitindo a alguém possa coagi-lo a votar num dado candidato. Como o eleitor diz a opção em voz alta, o atacante pode sempre verificar se o

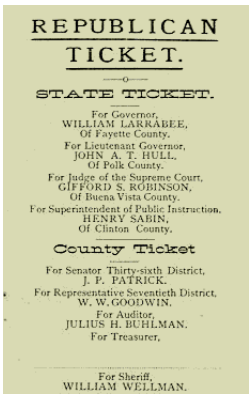
ataque foi bem sucedido. Este sistema não tem nenhum mecanismo que permita cumprir o requisito de anonimato e não coercibilidade. Outro dos grandes problemas deste sistema é o de o voto ser falado, o que pode levar a erros na sua anotação por parte dos responsáveis, perdendo assim, a precisão.

2.1.2 Papel

O sistema de voto em papel apareceu com o propósito de colmatar as falhas do sistema de voz, resolvendo o anonimato do eleitor e resolvendo, parcialmente, o problema da coação. Ele tornou-se o sistema mais usado nos dias de hoje, nos vários sectores da sociedade. Este sistema tem sido aperfeiçoado ao longo dos anos e, hoje em dia, é um dos sistemas mais seguros; no entanto, é o sistema mais caro de implementar, pois utiliza muitos recursos humanos e, mesmo assim, tem muitos problemas que serão detalhados no parágrafo seguinte.

Problemas

O primeiro problema que o sistema de voto em papel encontrou foi que, inicialmente, não eram dados boletins aos eleitores nas urnas. Os eleitores traziam-los consigo e, depois, introduziam-los na urna. Isto trouxe vários problemas desde casos onde o eleitor introduzia boletins errados, muitas vezes dados por atacantes para enganar o eleitor, pondo no boletim nomes de candidatos errados, ou com nomes de candidatos de várias listas diferentes no mesmo boletim de voto. Este último caso verificava-se quando o mesmo boletim servia para várias eleições. Um exemplo desses boletins é a Figura 2.2 .


REPUBLICAN
TICKET.

STATE TICKET.

For Governor,
WILLIAM LARRABEE,
Of Fayette County.
For Lieutenant Governor,
JOHN A. T. HULL,
Of Polk County.
For Judge of the Supreme Court,
GIFFORD S. ROBINSON,
Of Buena Vista County.
For Superintendent of Public Instruction,
HENRY SABIN,
Of Clinton County.

County Ticket

For Senator Thirty-sixth District,
J. P. PATRICK.
For Representative Seventieth District,
W. W. GOODWIN.
For Auditor,
JULIUS H. BUHLMAN.
For Treasurer,

For Sheriff,
WILLIAM WELLMAN.

Figura 2.2: Boletim de uma eleição americana em 1836[SB12]

Outro problema grave que acontecia era o fato de se descobrir em quem um eleitor votou, se fosse conhecido como era o seu boletim e se tivesse alguma característica única . Para acabar com este falha , os boletins hoje em dia são impressos e dados pelos responsáveis da eleição e são entregues no ato de votação. Este método foi introduzido na Austrália e por isso, tem o nome de boletim Australiano.

Esta forma de entrega de boletins, tornou o voto em papel mais seguro, mas como tem sido frequente ao longo da história, foi inventado outro tipo ataque a este sistema. Para tal funcionar é necessário que atacante tenha acesso a um dos boletins, antes da eleição, o que pode ser facilmente conseguido, visto que, da criação do boletim até ao preenchimento do voto na urna, os boletins passam por vários responsáveis e várias localizações. O atacante coage alguém a votar num dado candidato, dando o boletim preenchido ao eleitor, pede que o introduza na urna e que entregue o boletim que lhe é dado no ato de votar vazio. Este tipo de ataque é exequível nas eleições nacionais. Uma proposta para resolver este problema foi a introdução de um identificador único nos boletins, que posteriormente é verificado. Desta forma todos os boletins são únicos, o que permite ligar um boletim a um votante, algo que não é desejável nos sistemas de voto(em risco o anonimato).

Para além de ataques aos boletins, outro ataque bastante conhecidos foi utilização propositada de canetas com uma tinta especial, que desaparecia após algum tempo. Este ataque faria, por exemplo, que os votos escritos com essa caneta fossem todos votos em brancos. As canetas utilizadas nas eleições devem ser sempre auditadas para prevenir este ataque.

2.1.3 Eletrónico

O Voto eletrónico é a realização do voto através de equipamento eletrónico e a este sistema de voto dá-se o nome de **Sistemas de E-Voting**. Estes foram e continuam a ser muito estudados e, por isso, têm evoluído ao longo dos anos. Os modelos utilizados atualmente já se encontram muito robustos e precisos, o que possibilita que sejam usados em larga escala. Na Figura 2.3 é apresentado um desses modelos.

Figura 2.3: Maquina de voto eletrónico[SB12]

Problemas

Este sistema utiliza máquinas para retirar o erro humano no voto; no entanto, vários ataques foram inventados para interferir com este sistema. O que se veio a demonstrar é que estes sistemas, quando falham, têm um maior impacto na eleição, pois afetam um maior número de votos e não permitem recuperação. Inicialmente, o voto eletrónico era feito através de máquinas mecânicas, que foram um desastre, pois eram sistemas muito fáceis de adulterar, estragavam-se facilmente, podiam ficar encravadas e caso houvesse um erro no contador, toda a contagem na máquina estaria errada.

Com a revolução tecnológica, estes sistemas deixaram de ser mecânicos e passaram a ser controlados eletronicamente, utilizando, assim, software para resolver os problemas. As falhas encontradas neste sistemas podem ser divididas nestas 3 categorias

- Falha de software - O *software* têm falhas e tanto o voto eletrónico como o voto pela Internet necessitam de software. As falhas neste sistema podem fazer com que um atacante altere votos ou coloque a máquina num estado em que os votos não possam ser contados.
- Impossibilidade de recontagem - Caso haja uma falha grave, pode não ser possível fazer uma recontagem dos votos, sendo assim perdidos os votos realizados, sem possibilidade de voltar atrás na situação.
- Fraude em grande escala - Como as máquinas são feitas por humanos, é possível que um atacante se infiltre na produção do sistema e, por exemplo, ponha um código de modo a alterar os votos em grande escala.

2.1.4 Internet

Com a revolução da Internet, as pessoas começaram a utilizá-la para fazer atividades como compras e operações bancárias. No entanto, a votação pela Internet apresenta problemas específicos, difíceis de resolver. O I-Voting é o sistema de Voto Eletrónico que utiliza a Internet como o meio de transmitir o voto. Este sistema é a solução de voto que mais cresceu nos últimos anos. Por isso, é importante enumerar as suas vantagens e desvantagens. Com isto pretende-se analisar o custo/benefício de usar o sistema I-Voting. As vantagens inerentes ao I-Voting são [Oos04]:

- O custo por voto é baixo.
- O ato eleitoral pode ser realizado em qualquer local.
- O processo de contagem de votos é mais rápido, devido a ser realizada por computador.
- A contagem de votos é precisa, pois não depende de pessoas.
- Aumentar número de votos, não só por permitir que o eleitor de mobilidade reduzida possa votar, como também por permitir que eleitores afastados do local de voto, possam votar.

Problemas

As desvantagens trazidos pelo I-Voting são os ataques, que podem acontecer aos sistema de voto eletrónico, mais os ataques que se podem fazer pela Internet. De seguida vão ser enumeradas as desvantagens do I-Voting[Oos04]:

- Alteração de votos.
- Possibilidade de usar o voto de outro eleitor.
- Introdução de votos repetidos.
- Venda de votos.
- Falhas do sistema que causem indisponibilidade do serviço.

Para colmatar estas falhas é necessário implementar mecanismos de segurança. O foco desta tese passa por estudar os vários tópicos de segurança. Com eles, pretende-se selecionar as melhores soluções de segurança para implementar na aplicação. Por conseguinte, almeja-se alcançar todos os benefícios do I-Voting e mitigar ao máximo os seus riscos, garantindo sempre a segurança e legitimidade da votação.

2.2 Premissas criptográficas

A **Criptografia** é um termo oriundo da Grécia antiga e é o estudo de técnicas capazes de codificar uma mensagem e torná-la apenas legível para quem for detentor de um segredo. Com o uso de certos princípios que serão explicados nesta secção, é possível tornar uma mensagem ilegível para qualquer receptor que não possua o segredo.

A criptografia tem 4 objetivos principais:

- A **confidencialidade** da mensagem. Apenas quem tem a "chave secreta"deve conseguir ler o conteúdo da mensagem.
- **Integridade da mensagem**. A mensagem enviada deve ser exatamente igual à recebida.
- **Autenticação** do emissor da mensagem. Deve ser possível saber quem é o autor da mensagem.
- **Não-repúdio** do emissor da mensagem. O emissor não deve poder negar a autoria da mensagem.

A criptografia providencia segurança à informação, através de aplicações como cifra de mensagem, provas de conhecimento sem relevar o conteúdo, partilha de chaves e assinaturas digitais[AaAH11]. Estas aplicações da criptografia são essenciais para o voto pela Internet e, por isso, serão explicadas em mais detalhe nas secções seguintes.

A criptografia está dividida em duas categorias:

- De **chave simétrica/única/secreta** - Existe uma única chave que cifra e decifra a mensagem.

- De **chave assimétrica/pública/partilhada** - Existe uma chave para cifrar e existe outra chave diferente para decifrar.

2.2.1 Algoritmos de Chave Simétrica

Os algoritmos de chave simétrica são aqueles que utilizam para cifrar e decifrar mensagens uma única chave. Este tipo de criptografia é denominada, muitas vezes, com o nome de criptografia convencional e é a mais conhecida.

Na figura 2.4 é apresentado um esquema de como funcionam os algoritmos de Chave Simétrica. A figura contém um emissor da mensagem chamado Alice e um receptor chamado Bob. A Alice cifra a mensagem usando uma chave secreta. Depois envia a mensagem cifrada para o Bob. Ele recebe uma mensagem ilegível, pois está cifrada. Para a decifrar, o Bob utiliza a mesma chave secreta que a Alice. Com isto, caso a mensagem seja intercetada no envio, o atacante não conseguirá perceber o seu conteúdo.

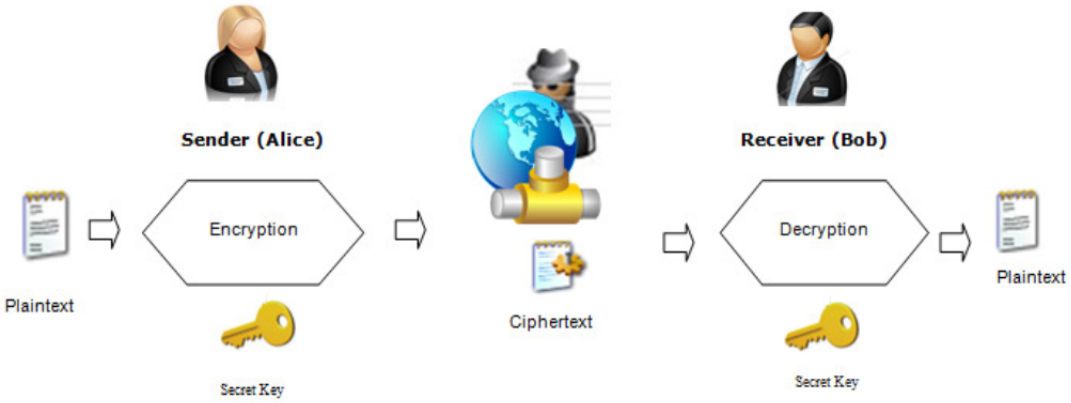


Figura 2.4: Sistema de Chave Simétrica[AaAH11]

Existem vários algoritmos de cifra de chave simétrica e podem ser usados, por exemplo, para cifrar conteúdo a ser enviado pela Internet ou guardado em disco.

2.2.1.1 DES

DES (Data Encryption Standard) - É um algoritmo que utiliza o modelo Fiestel Cipher e já foi quebrado por "força bruta". Usa uma chave de 56 bits e blocos de 64 bits. Neste momento já existe outra variante chamada **Triple DES** que o tornam muito mais seguro. Neste momento, existem algoritmos mais eficientes que os apresentados[Six11].

2.2.1.2 AES

AES (Advanced Encryption Standard) Algoritmo de cifra baseado no algoritmo de ***Rijndael***. Este algoritmo utiliza blocos de 128 bits e tem tamanhos de chaves 128, 192 e 256 bits. Este algoritmo é mais forte que o DES e tornou-se na cifra padrão. [Six11]

2.2.1.3 Serpent

Utiliza blocos de 128 bits e chaves de 128, 192 ou 256 bits. Este método tem uma desempenho muito fraco, mas é mais forte que o AES. No entanto, devido a demorar muito tempo a cifrar conteúdo, só é utilizado em casos onde a desempenho do algoritmo não seja relevante[SKW+00] (Nota: esta citação foi introduzida pelo autor na Wikipedia, pois faltava uma referência para o fato de o Serpent ser mais seguro e com pior desempenho que o AES).

2.2.1.4 Twofish

É a evolução do algoritmo Blowfish e passou de blocos de 64 bits (Blowfish) para 128 ou mais bits e as suas chaves variam entre 128, 192 e 256. Segundo o concurso da AES[SKW+00], esta opção é considerada uma combinação entre o AES e o Serpent, pois tem muita segurança e boa desempenho.

2.2.1.5 Threefish

É a evolução do algoritmo Twofish, permite chaves de 256 , 512 e 1024 bits e divide em blocos do mesmo tamanho. No entanto, várias falhas de segurança já foram encontradas neste sistema como o *boomerang attack*[BDK05] e *rotational attack* [KNR10].

Segundo o estudo *"Speedtest and Comparsion of Open-Source Cryptography Libraries and Compiler Flags"*[Bin08] o desempenho de cada um dos algoritmos criptográficos varia consoante o sistema operativo e biblioteca usada. A Figura 2.5, ilustra o resultado desse estudo.

Outro aspeto importante é o referenciado no "Cryptographic Misuse of Libraries" [DGKV14] onde se afirma que existem várias falhas de implementação nos algoritmos criptográficos e, por isso, as bibliotecas a ser utilizadas têm de ser analisadas em detalhe.

2.2.1.6 Limitações

Estes tipo algoritmos têm algumas limitações, como por exemplo, não conseguirem provar quem é o remetente da mensagem (propriedade do não-repúdio) e não conseguem providenciar quem é o destinatário. Por isso, estes algoritmos são usados para cifrar conteúdo.

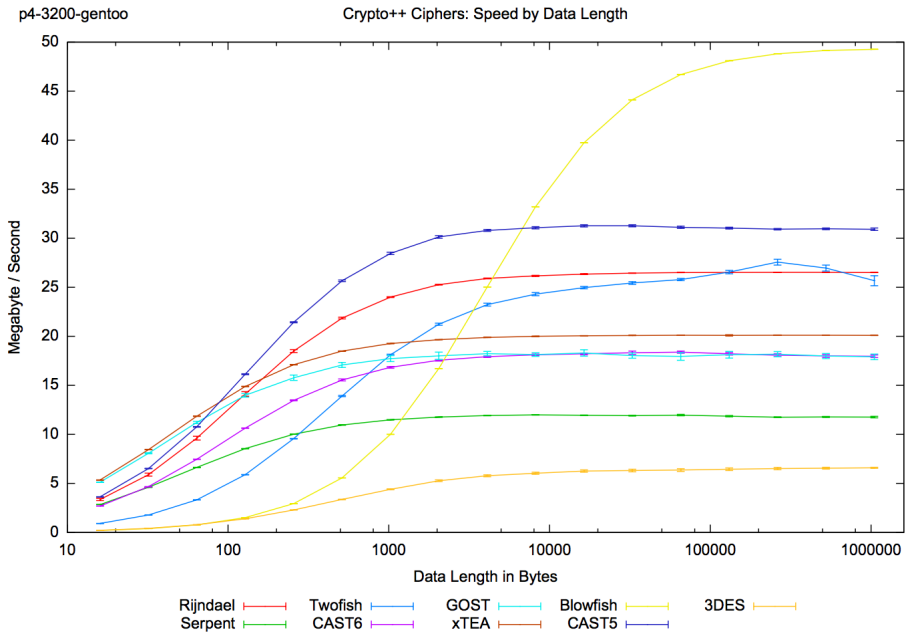


Figura 2.5: Comparação do desempenho dos algoritmos de cifra simétrica [Bin08]

2.2.2 Algoritmos de Chave Assimétrica

Os algoritmos de Chave Assimétrica diferem dos algoritmos de Chave Simétrica, porque utilizam uma chave para cifrar e outra para decifrar.

Na Figura 2.6 está representada uma das maneira como as chaves assimétricas funcionam. A Alice quer enviar uma mensagem ao Bob. Ela utiliza a chave pública do Bob para cifrar a mensagem. Como a mensagem fica cifrada, caso seja lida por um intruso, este não consegue compreender o seu conteúdo. O Bob partilha a sua chave pública, para que lhe possam enviar mensagens e para decifrar as mensagens recebidas, o Bob utiliza a sua chave privada.

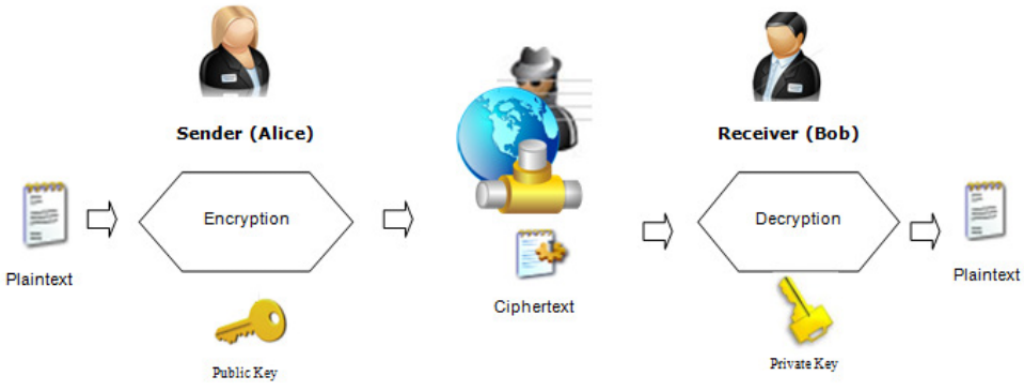


Figura 2.6: Sistema de Chave Assimétrica[AaAH11]

Usando este tipo de chave é possível estabelecer chaves seguras na Internet. A segurança deste sistema é baseada em problemas matemáticos, que são muito difíceis de resolver. Em princípio, é

possível decifrar qualquer mensagem cifrada utilizando força bruta, mas, no entanto, se não forem utilizados meios para reduzir o espaço de possíveis soluções, o tempo que levaria a decifrar a mensagem seria impraticável. Quanto maior o tamanho da chave, mais tempo é necessário para decifrar o conteúdo.

Existem atualmente 4 tipos de problemas matemáticos usados em cifras[AaAH11]:

- Problema de Fatorização de Inteiros - Este problema é o de encontrar divisores não triviais de um número muito grande. Este sistema é utilizado, pois não existe nenhum algoritmo eficiente para resolver o cálculo de divisores para números muito grandes. Neste sistema são utilizados dois números primos ao acaso, para o tornar o problema ainda mais complexo. A sua fórmula é dada pela Equação 2.1,

$$n = p.q \tag{2.1}$$

  em que p e q são dois números primos muito grandes. Este problema é usado por algoritmos, como o RSA.

- Problema Logarítmico Discreto - É o cálculo de um número K, em que (Equação 2.2):

$$b^k = a \tag{2.2}$$

  a e b, dois números reais. O seu cálculo é muito difícil e os seus casos médios são tão difíceis como os piores casos do problema de fatorização de inteiros. Um exemplo deste algoritmo é o ElGammal e consegue ter o mesmo nível de segurança que o RSA, utilizando chaves de menor dimensão[Gar14].

- Problema Logarítmico Discreto em curvas elípticas- É usado o problema de curvas elípticas(curva da função logarítmica) em espaços finitos. Foi proposto em 1985 por Neal Koblitz. A definição do problema das curvas elípticas é apresentado na Equação 2.3 e 2.4,

$$y^2 = x^3 + ax + b(mod p) \tag{2.3}$$

  onde a e b pertencem a um conjunto de números primos e

$$4a^3 = 27ab^2 \tag{2.4}$$

  de modo a que o polinómio não tenha raízes múltiplas. A qualidade desta cifra depende da curva: do seu gerador [LM10], do domínio e do tamanho da chave[Res00].

- Problemas baseados no Caos ou mapas do Caos - Estes problemas baseiam-se em sistemas caóticos, sistemas já muito estudados pela Física. Estes estão muito presentes no Universo. Neste tipo de problemas, uma pequena variação nos dados iniciais leva a resultados muito diferentes. Recentemente, o interesse neste tipo de cifras aumentou, porque possui duas

propriedades muito úteis para a criptografia: confusão e difusão. No entanto, ainda não existem implementações eficientes deste algoritmos [Ale14].

O Problema de Fatorização de Inteiros ganha em desempenho ao Problema Logarítmico Discreto, quando o tamanho das chaves é longo. No entanto, o Problema Logarítmico Discreto é mais difícil de quebrar que o Problema de Fatorização de Inteiros.

Dos problemas apresentados, o Problema Logarítmico Discreto em curvas elípticas permite uma segurança equivalente com chaves mais pequenas [HVM04]. Como tem uma melhor desempenho, este problema consome menos recursos que os outros, sendo uma opção mais acertada para dispositivos móveis.

De seguida, serão apresentados vários algoritmos de chave assimétrica que utilizam os problemas em cima referidos para tornar complexo a tentativa de descodificação.

RSA

O algoritmo RSA utiliza o problema de fatorização de número primos. Este algoritmo é muito utilizado, nas ligações HTTPS onde, servidor gera uma chave pública e uma privada e envia a chave publica para o cliente. O cliente, depois, gera uma chave simétrica, por exemplo, uma AES e, envia cifrando com a chave pública do servidor.

Este sistema também é utilizado em assinaturas digitais, porque permite que um utilizador com uma chave privada possa assinar uma dada mensagem. No sistema das assinaturas digitais é utilizada a *Hash* da mensagem e não o seu conteúdo por uma questão de desempenho. Na Figura 2.7 está representado um diagrama que representa como é feita uma assinatura digital.

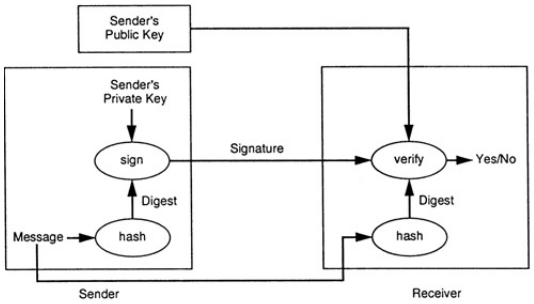


Figura 2.7: Assinatura digital[KT06]

A segurança do RSA depende do tamanho da chave escolhida e tem vários problemas como por exemplo[Gar14]:

- A geração de chaves é lenta.
- Lento a cifrar texto.
- Tem de iniciar com dois números primos grandes.

ElGamal

A cifra ElGamal é uma aplicação direta do algoritmo de troca de chaves Diffie-Hellman e utiliza o Problema Logarítmico Discreto. A diferença entre o algoritmo ElGamal e o Diffie-Hellman é que o ElGamal é semanticamente seguro e o Diffie-Hellman não[EAEZ14], pois é seguro contra "ataques de texto cifrado"[ElG85]. O ElGamal utiliza encriptação probabilística garantindo que a encriptação de duas mensagens iguais não dá o mesmo resultado. O ElGamal pode ser utilizado para assinatura digital.

Sistemas usando Problema Logarítmico Discreto em curvas elípticas

Os sistemas que usam Problema Logarítmico Discreto em curvas elípticas têm um maior desempenho computacional e usam menos largura de banda. Utilizam chaves mais pequenas que os outros sistemas[EAEZ14].

2.2.3 Outros Princípios Criptográficos aplicados ao voto

Nesta secção serão analisados princípios criptográficos, que são úteis para o voto electrónico.

Cifra Homomórfica

A cifra homomórfica é uma forma de cifrar os dados, que através de um operação feita aos dados cifrados, permite extrapolar os dados decifrados sem nunca realizar a decifra. Este conceito é muito útil para filtrar e reencaminhar dados cifrados consoante o destinatário da mensagem. No caso do voto eletrónico, permite saber a intenção de voto de um eleitor, sem nunca o decifrar o voto. A computação em nuvem é uma área onde estes modelos são muito usados como método de corresponder tráfego para o seu destinatário. Em suma, a cifra homomórfica serve para fazer correspondências.

Esta cifra baseia-se nos modelos homomórficos, que são sistemas criptográficos que preservam operações, como por exemplo, a de multiplicação ou a de adição de blocos.

Novos modelos homomórficos foram descobertos em 2012, permitindo dar novos usos a este tipo cifra[GGI$^+$14].

No caso do voto, este sistema permite ligar o votante ao voto, sem comprometer a sua identidade e utiliza "segredo partilhado", como meio para garantir do anonimato.

Provas sem conhecimento

A cifra homomórfica permite redirecionar o tráfego para um destinatário ou, aplicando ao caso do voto, verificar se um voto pertence a um certo candidato no ato da contagem. No entanto, não permite ao remetente confirmar se a mensagem foi lida corretamente, ou no caso da contagem de votos, verificar se o voto do eleitor foi realmente contado.

Através das provas sem conhecimento é possível ao eleitor confirmar que o seu voto foi realmente contado. É um método usado na criptografia e é utilizado para provar que se conhece algo sem ser obrigado relevar o seu conteúdo[Bri12].

No caso do voto com cifra homomórfica é utilizado o método não interativo das provas de conhecimento, que através de uma aritmética permite ao votante realmente confirmar o seu voto foi contado corretamente. Este sistema usa chaves assimétricas e precisa de uma "ajuda"de uma autoridade confiável[GGI+14].

Assinatura cega

A assinatura cega é uma forma de assinatura digital, onde a mensagem é assinada sem verificar o conteúdo da mensagem. Este método permite que, num sistema de votação, o voto seja totalmente dissociado do eleitor, mantendo audibilidade do sistema e o anonimato. Para o utilizar é necessário existir um componente que torne os votos anónimos, assinando-os, sem nunca verificar o seu conteúdo. Neste caso, não existe ligação entre o voto e o eleitor. Este sistema pode ser implementado usando o RSA [Cha84].

Canais anónimos

Os canais anónimos foram introduzidos por David Chaum's e permitem criar eleições anónimas, moeda digital e até criar redes seguras como o TOR [DMS04]. Neste sistema a comunicação é feita por camadas ou cadeias. Cada camada recebe uma lista de mensagens cifradas que são parcialmente decifradas, para saber quem é o próximo destinatário, sendo depois cifradas e reordenadas para serem enviadas para a próxima camada[ALBD04].

Segredo partilhado

O segredo partilhado é uma maneira de cifrar conteúdo com uma chave pública e para decifrar, gera uma chave, que é partida em X pedaços e que necessita de Y pedaços para ser reconstruida. Este sistema é útil para cifrar o voto, pois caso o voto seja cifrado com essa chave, ele só será decifrado quando um Y número de supervisores o fizer. Se Y for maior que um, um supervisor sozinho não consegue decifrar um voto[Pei12].

2.3 Sistemas de votação pela Internet

Os sistemas de E-Voting pertencem a um destes 4 modelos[Bri12]:

1. **Modelo de Canais Anónimos** - Neste modelo é possível o voto anónimo. Ele serve-se de várias *camadas* para garantir o anonimato. Cada vez que o voto passa por uma *camada* são

geradas chaves. Por isso, este método traz anonimato mas requer poder muito computacional.

2. **Modelo de Assinatura Cega** - Este método usa as assinaturas cegas para tornar o voto anónimo, dissociando o voto do eleitor.

3. **Modelo de Benaloh** - Neste modelo cada fração do voto é partilhada com várias autoridades de voto. Este esquema necessita de uma boa ligação, pois tem custos elevados de comunicação. Utiliza o princípio do segredo partilhado.

4. **Modelo de Cifra Homomórfica** - Aplica as propriedades das funções homomórficas para calcular o resultado da eleição, sem ter de decifrar os votos.

Já existem vários sistemas que permitem votar pela Internet. Neste capítulo será feita uma análise a cada um dos sistemas existentes.

Sistemas usados na Suíça [Bri12]

Os sistemas Suíços utilizam a Internet para votar. Eles usam um sítio com ligação HTTPS para exercer o seu voto. Consoante o Cantão Suíço, os métodos de autenticação são diferentes, como por exemplo, PIN enviado para o correio ou palavra passe.

Sistema usado na Estónia [SFD+14]

Na Estónia, a Internet é uma das maneiras de votar. Eles utilizam um cartão de identificação para se autenticarem na página da Internet. Este sistema tem 2 particularidades interessantes que são: permitir que o voto seja alterado durante o período da votação e que, em caso de falha, já existem leis para resolver a situação. Recentemente, lançaram aplicações de voto para IOS e Android. Na secção seguinte é apresentada a versão para smartphones.

Helios [BGP11]

O sistema Helios permite o voto na Internet. Garante privacidade e anonimato e é uma ferramenta *Open-Source*.

As premissas criptográficas que aplica o modelo homomórfico e o algoritmo ElGamal .

SureVote [Cha01]

Utiliza o modelo de canais anónimos, mas sofre de várias limitações, como a impossibilidade rastrear o voto e não permitir verificar se o voto foi contado corretamente.

REVS [JZF03]

Sistema que serve-se de vários servidores e é resistente a falhas. Aplica assinaturas cegas para confirmação, usa chaves partilhadas no processo de votação e canais anónimos, como método para contagem de votos.

MobileREVS [LC06]

Este sistema foi desenhado para o voto eletrónico via Internet e GMS, em particular *smartphones*. O seu sistema pode ser usado para eleições, sondagens e referendos.

O MobileREVS utiliza a biblioteca *Bouncy Castle* para assegurar a integridade dos votos e o anonimato dos eleitores. Para criar assinaturas aplica o RSA, com uma chave de 1024 bits e SHA-1 para assinar a mensagem. A cifra simétrica usada é o *Triple-DES*.

SCV [KZ07]

Conta com um sistema de canais anónimos para garantir o anonimato com assinaturas cegas.

EVIV [Bri12]

O EVIV é um sistema de Voto que aplica um modelo de *tokens* e de ligação seguras ponto a ponto, para garantir as propriedades do voto.

2.4 Sistemas de voto através de smartphones

Já existem algumas abordagens para o voto na Internet através de *smartphone* e nesta secção serão expostas essas abordagens. Existem aplicações de Voto pela Internet no GooglePlay, no entanto estas aplicações não utilizam ligações seguras e usam servidores de configuração desconhecida. Por isso, as aplicações não documentadas do Google Play não serão analisadas.

Nesta secção, serão apresentadas as aplicações móveis de voto. É importante salientar que nem todas as aplicações aqui apresentadas utilizam a Internet como meio de comunicação, no entanto, foram consideradas importantes de analisar, pois, têm outras características semelhantes.

VK [VK13]

O VK é a aplicação móvel lançada pelo governo da Estónia para seu o sistema de voto, que contempla votação por computador e *smartphone*. Eles disponibilizaram, através do GitHub, o código fonte da suas aplicações. Este sistema usa TLS e leitor de QR. De todas as aplicações apresentadas esta é a única utilizada em eleições reais e, por isso, foi utilizada como métrica de comparação com a solução a desenvolver. Como tem o seu código fonte aberto, foi possível analisar em detalhe os mecanismos de segurança implementados pela mesma. A linguagem utilizada para comunicar é o XML.

A+Votz [Rag14]
Segundo os autores da aplicação [Rag14], o A+Votz foi desenvolvida usando a tecnologia Android. Ela utiliza como meio de autenticação a impressão digital, permitindo assim, identificar o eleitor inequivocamente. Esta abordagem permite proteger o voto em caso de roubo do telemóvel ou descoberta dos dados de autenticação. A aplicação ainda utiliza *Assisted-GPS* para guardar a localização onde o voto foi realizado. Apoia-se em computação em núvem. O código não é aberto, logo as implementações de segurança não serão analisadas.

Biométricas Foi utilizado um esquema com duas impressões digitais sobrepostas para criar um identificador da impressão digital e foi utilizado o algoritmo *BAYESIAN DECISION FUSION* para encontrar a impressão digital na base de dados. Este sistema permite uma adaptação ao erro do leitor de impressão digital que diminui os custos.

Assisted-GPS O mecanismo "Assisted-GPS"é um sistema de triangulação por rede telemóvel, que é a prova de ataques de *spoofing*. Esta solução foi usada para resolver os problemas do GPS quando não tem sinal em locais fechados.

Esteganografia O sistema do A+Votz usa esteganografia para enviar mensagens secretas através de imagem. Este sistema é pouco seguro contra ataques que procurem relevar as mensagens escondidas nas imagens. Para contornar o problema, o autor apresenta duas soluções. A primeira solução é tornar mais difícil a detenção dos bits usados na mensagem, alterando bits para além dos LBS (sistema tradicional de estenografia). A segunda solução é colocar a mensagem em camadas de nível mais baixo da imagem.

Princípio Criptográfico É utilizado o modelo Homomórfico para a contagem.

Boardroom voting [evB14]
O Boardroom Voting é uma aplicação de voto eletrónico que serve para realizar votos entre um grupo de pessoas na mesma localização. Os princípios Criptográficos usados não se aplicam à solução que irá ser desenvolvida para esta dissertação, porque não existe um servidor central, que é um requisito obrigatório.

Decentralized E-Voting on Android Devices Using Homomorphic Tallying [Rit14]
Segundo o autor, o sistema apresentado é um modelo descentralizado de voto, isto é, não existe um servidor central. É um protocolo menos eficiente do que o Boardroom voting apresentado na parágrafo anterior, porque esta aplicação utiliza um protocolo, que não foi desenhado inicialmente para ser implementando em servidores descentralizados.

Princípio Criptográfico Utiliza cifra simétricas e modelo o Homomórfico.

E-Voting System Using Android Application [MKRS14]
Segundo o autor do artigo "*E-Voting System Using Android Application*", ele aplica o conceito dos QR-Code (um código de barras de duas dimensões) para fazer autenticação de um utilizador. Com este sistema, é possível decifrar de forma rápida o QR-code de identificação do eleitor. Permite o voto feito através de SMS ou pela Internet.

Princípio Criptográfico Utiliza a esteganografia para cifrar mensagens.

Implementation of Electronic Voting System in Mobile Phones with Android Operation System [VGgTG13]
Segundo o autor do artigo "*Implementation of Electronic Voting System in Mobile Phones with Android Operation System*", este sistema usa a Internet para votar e permite a confirmação da contagem do voto.

Princípio Criptográfico Para assinaturas cegas é usado o RSA e para cifrar dados em disco é usado o AES. O SHA-256 é a função de *Hash* utilizada para autenticação.

Android Powered Portable Voting Device [Mur11]
Este sistema permite o voto eletrónico pela Internet. Não é especificado o modelo de SVE.

Princípio Criptográfico Utiliza protocolo HTTP e SSL para comunicar com o servidor e vice-versa. A linguagem utilizada para comunicar é o XML. O TLS foi implementado e foi utilizado o OpenSSL para gerar certificados. A cifra simétrica que utiliza é o DES para cifrar o voto.

2.5 Sistema operativo em dispositivos móveis

Nesta seção são abordadas os dois sistemas operativos mais usados em dispositivos móveis: o Android e o IOS.

2.5.1 Android

Em 2014, o sistema operativo Android foi o mais utilizados em *smartphone*. O seu código foi escrito em C e C++. As aplicações que correm no Android são escritas em Java e executadas em ambiente de máquina virtual. O nome da máquina é **Dalvik Virtual Machine** (DVM). O DVM

está otimizado para correr em ambientes, onde existe pouco poder de processamento e pouca memória[VGgTG13], como é o caso dos *smartphones*.

Sempre que uma aplicação é executada no Android, ela tem a sua própria máquina virtual isolada e tem o seu próprio espaço de memória, permissões e identificação.

Arquitetura

O Android pode ser dividido em 4 camadas apresentadas na Figura 2.8, que são o Linux Kernel e ferramentas de baixo nível, Biblioteca do sistema e Android Runtime, a ferramenta das Aplicações e uma camada para as aplicações[SPDS14]:

- Linux Kernel - É uma camada de ligação entre o *hardware* e o *software*. O Kernel usado no Android tem modificações implementadas pela Google. O Kernel é a camada responsável pela gestão de processos essenciais ao sistema operativo, como a gestão de drivers, processos e memória.
- Biblioteca do Sistema e Android Runtime - As Biblioteca do Sistema são funções escritas, usando a linguagem de programação C e C++, que são usadas pelo hardware do dispositivo. O SQLite, OPENGL e SSL são algumas das bibliotecas presentes nesta camada. O Android Runtime está implementado no DVM e algumas bibliotecas principais do JAVA.
- Plataforma de Aplicações - Nesta camada encontram-se alguns gestores do telemóvel, como o gestor de chamadas, notificação, rede e atividade.
- Aplicações - É a camada mais usada pelo utilizador e contém as aplicações que correm no dispositivo. Elas são desenvolvidas por programadores e são escritas usando a linguagem JAVA.

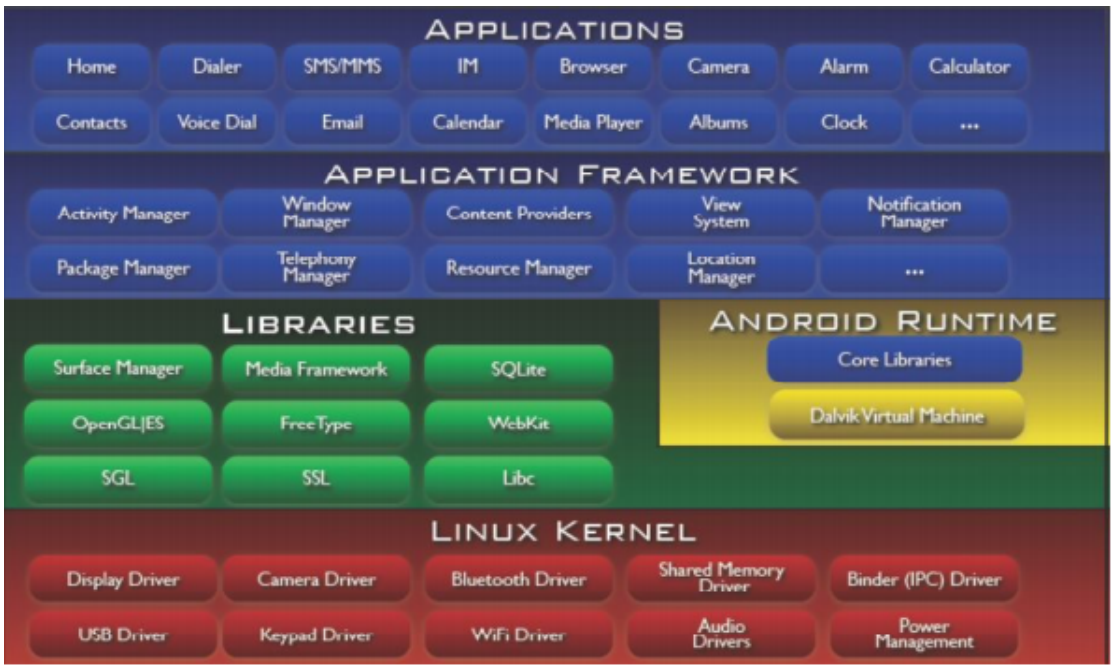


Figura 2.8: Arquitetura do Android[And14]

Mecanismos de segurança
Mecanismo de segurança implementados no Android:

- Mecanismos que lidam com o isolamento e segurança.
- Cada aplicativo é executado no seu próprio espaço isolado, com utilizador único e identificadores de grupo.
- As aplicações não estão autorizadas a trocar dados, a menos que solicitem permissões do utilizador.

2.5.2 IOS

IOS é um sistema operacional móvel da Apple Inc. desenvolvido originalmente para o iPhone. Neste momento, também é usado em iPod touch, iPad e Apple TV. A empresa Apple não permite que o seu sistema operativo seja usado em hardware de terceiros. Ao contrário do Android, este software não é *Open-Source*.

O sistema operativo pertence à família dos MAC OS/Unix-like.

Arquitetura

O IOS pode ser dividido em 4 camadas apresentadas na Figura 2.9. As camadas são:

- Core OS - Nesta camada estão presentes as interfaces UNIX, POSIX e o sistema de ficheiros baseados em C. Esta camada está em escrita em C e é a camada de mais baixo nível.
- Core Services - Nesta camada residem as interfaces que permitem programar Orientado a Objetos e o SQLite.
- Media - Contém bibliotecas como o OPENGL , Core Áudio e animações.
- Cocoa Touch - É a camada de mais alto nível, escrita em objetive-C ou Swift e é onde se situam as aplicações.

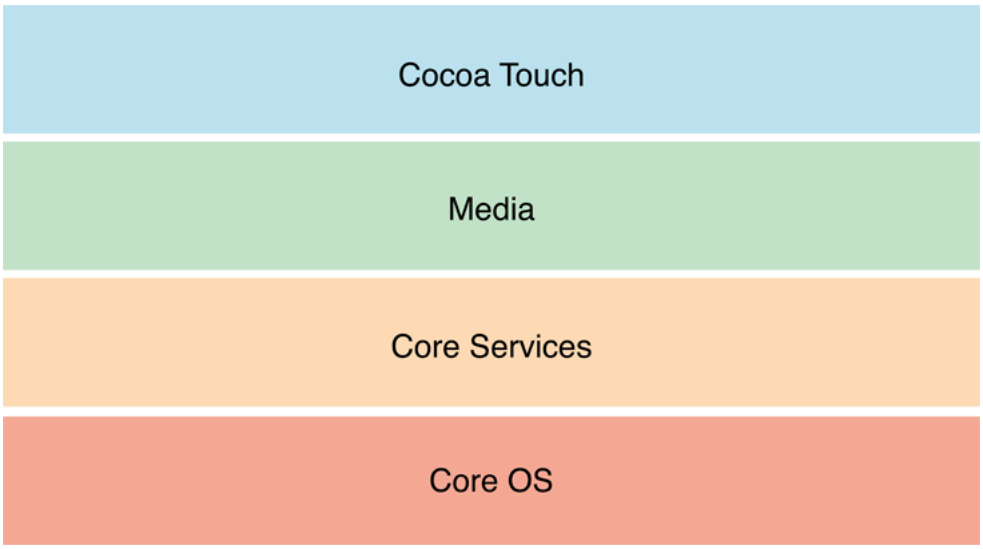


Figura 2.9: Arquitetura do IOS[Inc14]

Mecanismo de segurança
Mecanismo de segurança implementados no IOS:

- DEP (data execution prevention) - Este mecanismo torna difícil distinguir entre código e dados.
- ASLR (Address space layout randomization) - Usa endereços de forma aleatória, o que torna muito difícil o ato de encontrar informação em disco.

2.5.3 Tipos de ataques ao Android e IOS

Os tipos de ataques a *smartphones* podem serem divididos em 4 tipos[Mur11].

Roubo do telemóvel Com este tipo de ataque é possível retirar as informações não protegidas do telemóvel. A cifragem do telemóvel mitiga este tipo de risco.

Ataques *Man in the middle* Este caso acontece quando o atacante tem acesso ao canal de comunicação. Neste caso é a Internet. O atacante pode estar ligado à mesma rede que o utilizador ou ter acesso a um router por onde a comunicação passa. Para combater este tipo de ataques utiliza-se protocolos como SSL/TLS e assinatura digital.

Software malicioso Estas aplicações tentam replicar uma aplicação legítima, de modo a tentar aceder aos dados do utilizador ou então, roubar dados de outras aplicações.

Acesso *root* Smartphones com *root* estão mais vulneráveis, porque permitem que uma aplicação corra como administrador. Em *smartphones* sem *root*, uma aplicação descarregada nunca corre como administrador, exceto se for um software malicioso que explore uma falha crítica de um sistema e ganhe *root*.

2.6 Métricas de desempenho e segurança

Uma das métricas usadas para comparar as várias aplicações móveis de voto é o desempenho das aplicação [AaAH11].

De modo a testar a aplicação serão feitas análises à memória e serão feitos ataques à solução e ao sistema de voto, com o intuito de testar a sua segurança. Os ataques e testes a realizar foram retirados do **OWASP**[Pro14] (Open Web Application Security Project). Na Figura 2.10 é nos apresentado as 10 maiores ameaças a aplicações móveis.

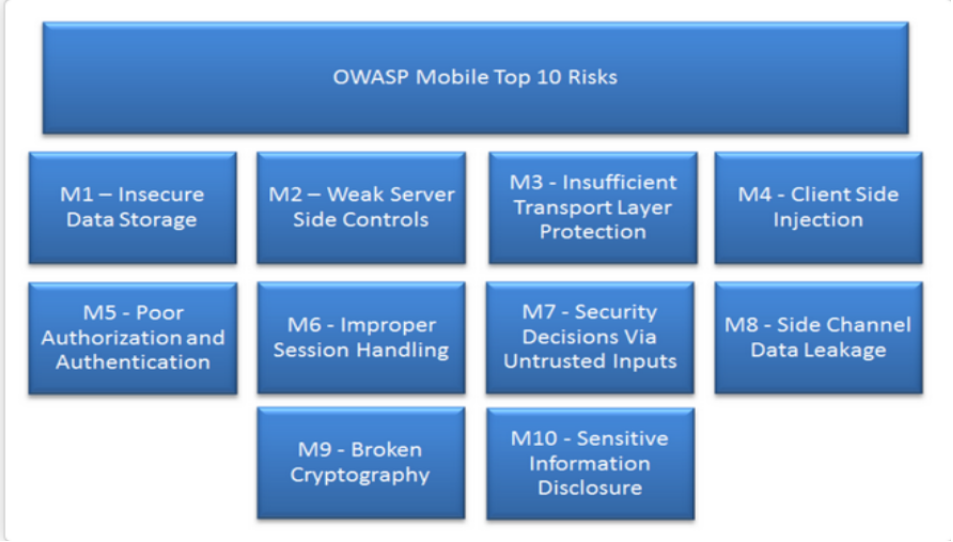


Figura 2.10: Top 10 de riscos em Mobile[Pro14]

2.7 Conclusões

Com a análise deste capítulo foi possível compreender que todos os sistemas de voto têm problemas e que a criptografia está na base dos sistemas de voto pela Internet. Foram descritas as várias cifras simétricas e assimétricas, assim como os princípios criptográficos, que podem ser aplicados ao voto: as cifras homomórficas, provas sem conhecimento, assinaturas cegas, canais anónimos e segredo partilhado.

Foi feita uma recolha dos sistemas de voto pela Internet já existentes, assim, como as que utilizam aplicações móveis. Por último, foi feita uma análise aos sistemas operativos Android e IOS e foram estudadas algumas métricas de segurança.

Capítulo 3

Descrição e projeto

O CertVote é a nova geração do voto eletrónico da Multicert que foi introduzida no mercado no final do ano 2014 e utiliza um modelo criptográfico para garantir que os requisitos do voto, apresentados no início do capítulo anterior, são cumpridos.

O antigo sistema de voto utilizava uma *applet* Java para votar que requeria ao utilizador a instalação de um *plugin* para exercer o voto. A tecnologia Java *applet* tem diversas falhas conhecidas e tem problemas de compatibilidade com vários navegadores e diferentes JVM (*Java Virtual Machine*). Com o intenção de oferecer uma solução mais fácil de utilizar, o CertVote foi construído usando as ferramentas HTML5, CSS3 e Javascript para a interface do utilizador e usa Java com a *framework* Spring para os servidores. Este sistema permite oferecer uma experiência de utilização muito mais rápida e "amigável". Está equipado com muitas novas funcionalidades como por exemplo:

- Componente de Caderno Eleitoral, que permite gerir o processo de votação por meio eletrónico ou presencial.
- Componente de Voto, que propicia aos utilizadores exercerem o seu voto, através de uma interface web, onde são apresentados os boletins e as escolhas feitas e é efetuada a submissão do voto.
- Componente de Resultados, que suporta o encerramento da votação, contagem de votos e apresentação de resultados.
- Autenticação com credenciais dinâmicas que oportuniza, a cada eleição, ter credenciais diferentes; por exemplo, numa eleição pode ter como credenciais o CC, NIF e senha enquanto que noutra eleição pode o ser *email* e uma senha.
- Envio de senha através de *email* ou *SMS* .
- Sistema de Auditoria que regista certas ações, como por exemplo, a de um utilizador se autenticar ou votar, permitindo assim que auditores consigam analisar as interações do sistema com os utilizadores. Estes registos foram cuidadosamente realizados com o intuito de nunca divulgar informação sobre a identidade de cada utilizador.
- Escolha de Domínio de Votação, proporciona que as eleições usem domínios exteriores à Multicert e continuar a usar o CertVote.

- Utilização de Chave de Eleição Partilhada que é um modelo onde todos os votos são cifrados por uma chave-mãe. A chave-mãe é uma chave que é dividida em um M conjunto de chaves-filhas e apenas necessita de N (menor ou igual a M) chaves-filhas para ser gerada. Este processo é realizado através do algoritmo "chave partilhada".

O problema proposto, a ser resolvido nesta dissertação, foi a criação e implementação de um sistema que interagisse com o CertVote já existente e permitisse aos eleitores votar com segurança através de *smartphones*, dando assim, aos seus eleitores uma maneira de exercerem o seu direito de voto mais rápido, cómodo, em qualquer lugar e em segurança.

Para criar esta solução foi necessário desenvolver vários componentes; entre eles, um servidor, que serve de eixo de ligação entre as aplicações móveis e o CertVote e um componente móvel. Várias opções foram tomadas para obter a compatibilidade entre os vários sistemas.

Este capítulo está dividido em três secções. Na primeira secção são expostos os requisitos funcionais e não funcionais da solução, assim como os atores do sistema, narrativas e casos de utilização e, por último, um *project charter* onde é possível visualizar uma maqueta do resultado final da aplicação. Na segunda secção é feita a apresentação da arquitetura proposta; é demonstrada uma visão geral de todo o sistema e , de seguida, são apresentados os ecrãs de navegação, a arquitetura física e a arquitetura do modelo criptográfico usado no CertVote; é ainda descrito em detalhe o protocolo de comunicação. Na última secção é feito um estudo de segurança, onde se define o que se pretende proteger e quem são os possíveis atacantes, sendo de seguida apresentados os pontos de ataque e os caminhos por onde um atacante pode adulterar o sistema.

3.1 Requisitos

Esta secção tem como finalidade estabelecer os principais objetivos para o projeto, assim como, enumerar e especificar os requisitos que o produto final deve satisfazer, de acordo com o que foi acordado com a Multicert. Com isto, pretende-se criar uma especificação, que permite comparar a solução apresentada com as necessidades do cliente. Para além disso, ainda são mostrados os atores que vão interagir com o sistema desenvolvido, bem como, são especificadas as narrativas e casos de utilização. Por último, é ostentado o *project charter*.

3.1.1 Requisitos funcionais

Os requisitos funcionais são objetivos que os vários componentes do sistema têm de cumprir. De seguida, serão expostos os requisitos do servidor e da aplicação.

Servidor

O servidor é o componente do sistema que vai comunicar com o CertVote e com o *smartphone*. Os seus requisitos estão apresentados na Tabela 3.1:

Tabela 3.1: Requisitos do servidor

ID	NOME	DESCRIÇÃO	PRIORIDADE
R01.01	Eleições em simultâneo	O servidor tem de ser capaz de suportar várias eleições em simultâneo	Alta
R01.02	Encontrar a eleição	O servidor tem de conceder que o utilizador possa encontrar a eleição em que pretende votar	Alta
R01.03	Encontrar a eleição por QR-CODE	O servidor tem de admitir que um utilizador utilize um código QR-CODE para encontrar a eleição	Baixa
R01.04	Autenticação	O servidor tem es estar preparado para verificar se o utilizador tem as credenciais corretas, para uma dada eleição	Alta
R01.05	Entrega de boletim	O servidor tem de entregar os boletins corretos ao eleitor	Alta
R01.06	Submeter voto	O servidor tem de submeter o voto do eleitor corretamente	Alta
R01.07	Guardar registos	O servidor tem de ser capaz de guardar registos para proporcionar auditorias	Alta

Aplicação

Os requisitos que a aplicação móvel tem de cumprir estão descritos na Tabela 3.2.

Tabela 3.2: Requisitos da aplicação

ID	NOME	DESCRIÇÃO	PRIORIDADE
R02.01	Encontrar a eleição por código	A aplicação tem de ser capaz de encontrar uma eleição através de um código	Alta
R02.02	Encontrar a eleição por QR-CODE	A aplicação tem de ser capaz de encontrar uma eleição através de um QR-CODE	Baixa
R02.03	Autenticação com credenciais dinâmicas	A aplicação tem de ser adaptável, pois as credenciais variam entre eleições	Alta
R02.04	Apresentação dos boletins	A aplicação tem de permitir que o utilizador visualize os boletins	Alta

R02.05	Escolha das opções	A aplicação tem de permitir que o utilizador escolha uma, nenhuma ou várias opções de voto	Alta
R02.06	Apresentação das opções escolhidas	A aplicação tem de apresentar as opções escolhidas antes de permitir submeter o voto	Alta
R02.07	Passagem por todos os boletins	A aplicação tem de obrigar o utilizador a percorrer todos os boletins para votar	Alta
R02.08	Aplicação de prazos	A aplicação tem de voltar ao ecrã de escolha de eleição após um certo tempo de ter começado o ato de preencher o boletim	Baixa
R02.09	Limpar dados em 2º plano	A aplicação tem de voltar ao ecrã de escolha de eleição caso a aplicação seja posta em 2º plano	Baixa
R02.10	Limpar dados após votação	A aplicação tem de garantir que após o voto toda a informação da sessão é removida do sistema	Alta
R02.11	Não guardar dados	A aplicação não pode guardar dados do utilizador	Alta

3.1.2 Requisitos não funcionais

Os requisitos não funcionais relacionam-se com o uso da aplicação em termos de vários critérios, como por exemplo, de segurança, disponibilidade e desempenho. Os requisitos a seguir expostos são os considerados mais importantes:

- RNF 1 - O sistema deve garantir a proteção dos dados a acessos não autorizados.
- RNF 2 - O sistema deverá estar disponível 99,9 porcento durante o tempo da eleição.
- RNF 3 - O servidor deverá operar com qualquer sistema operativo.
- RNF 4 - O servidor deverá ser escrito em Java e usar a *framework* Spring.

3.1.3 Atores

Os atores presentes no sistema são divididos em dois tipos que estão definidos na Tabela 3.3

NOME	PERMISSÕES
Visitante	Utilizador que pode introduzir código da eleição, ou apontar para o QR-Code e pode fazer login
Eleitor	Utilizador que pode ver os boletins de voto, preenchê-los e submeter o voto

Tabela 3.3: Atores

3.1.4 Narrativas de utilização

Para o sistema proposto, consideram-se as seguintes narrativas de utilização, com o esforço estimado usando a escala de Fibonacci, prioridade, caso de uso e ecrãs utilizados.

Tabela 3.4: Narrativas de utilização

ID	NOME	DESCRIÇÃO	PRIORIDADE	ESFORÇO	UCS	ECRÃS
US01	Usar QR-CODE	Como visitante, quero utilizar o meu QR-Code da eleição para a encontrar	Baixa	3	UC02	1
US02	Introduzir código da votação	Como visitante, quero introduzir o código da eleição para a encontrar	Alta	2	UC01	1
US03	Introduzir credenciais	Como visitante, quero introduzir as credencias e autenticar	Alta	5	UC03	2
US04	Preencher boletins	Como Eleitor, quero escolher as minhas opções de voto nos boletins	Alta	8	UC04	3
US05	Ver ecrã de resultados	Como Eleitor, quero ver as opções que escolhi em cada boletim	Alta	2	UC05	4
US06	Submeter voto	Como Eleitor, quero submeter o meu voto	Alta	8	UC06	4

3.1.5 Casos de Utilização

O modelo de casos de utilização permite relacionar os atores existentes com as ações que estes podem realizar. É uma representação externa e de alto nível do sistema: representa as funcionalidades do produto acessíveis ao utilizador final. Os casos de utilização estão detalhados no apêndice 1A .

IDENTIFICADOR	**UC01**
NOME	Usar QR-CODE
DESCRIÇÃO SUMÁRIA	Permite ao visitante utilizar a camera do telemóvel e um código QR-CODE para entrar na página de credenciais
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet.
PÓS-CONDIÇÕES	O visitante entra no ecrã de credenciais
PROCEDIMENTO	1. Carregar no botão "Usar QR-CODE" 2. Apontar câmera para o código

Tabela 3.5: Caso de utilização UC01

IDENTIFICADOR	**UC02**
NOME	Introduzir código
DESCRIÇÃO SUMÁRIA	Permite ao visitante introduzir um código único para entrar na eleição
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet.
PÓS-CONDIÇÕES	O visitante entra no ecrã de credenciais
PROCEDIMENTO	1. Preencher campo de texto. 2. Carregar no botão "Submeter código".

Tabela 3.6: Caso de utilização UC02

IDENTIFICADOR	**UC03**
NOME	Introduzir credenciais
DESCRIÇÃO SUMÁRIA	Permite ao eleitor aceder ao seu boletim de voto
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet e escolhido uma votação aberta.
PÓS-CONDIÇÕES	O eleitor entra no ecrã no ecrã de preencher boletins
PROCEDIMENTO	1. Preencher campos de texto. 2. Carregar no botão "Submeter credenciais".

Tabela 3.7: Caso de utilização UC03

3.1.6 Project Charter

Ecrã 1 - Entrar na eleição

Ecrã 2 - Autenticar

Ecrã 3 - Escolha de candidato

Ecrã 4 - Submeter o voto

Figura 3.1: Project Charter

3.2 Arquitetura

Com esta secção pretende-se especificar e apresentar o resultado de toda a arquitetura desenvolvida, definindo assim todas as camadas do sistema. A arquitetura foi desenvolvida com o intuito de cumprir os requisitos propostos e com vista a criar a melhor abordagem para resolver o desafio almejado. Na arquitetura será mostrada uma visão geral do sistema, a navegação na aplicação, as várias arquiteturas utilizadas e, por último, o protocolo de comunicação.

3.2.1 Visão geral

A visão geral serve para mostrar o essencial, a alto nível, do sistema e do seu meio envolvente.

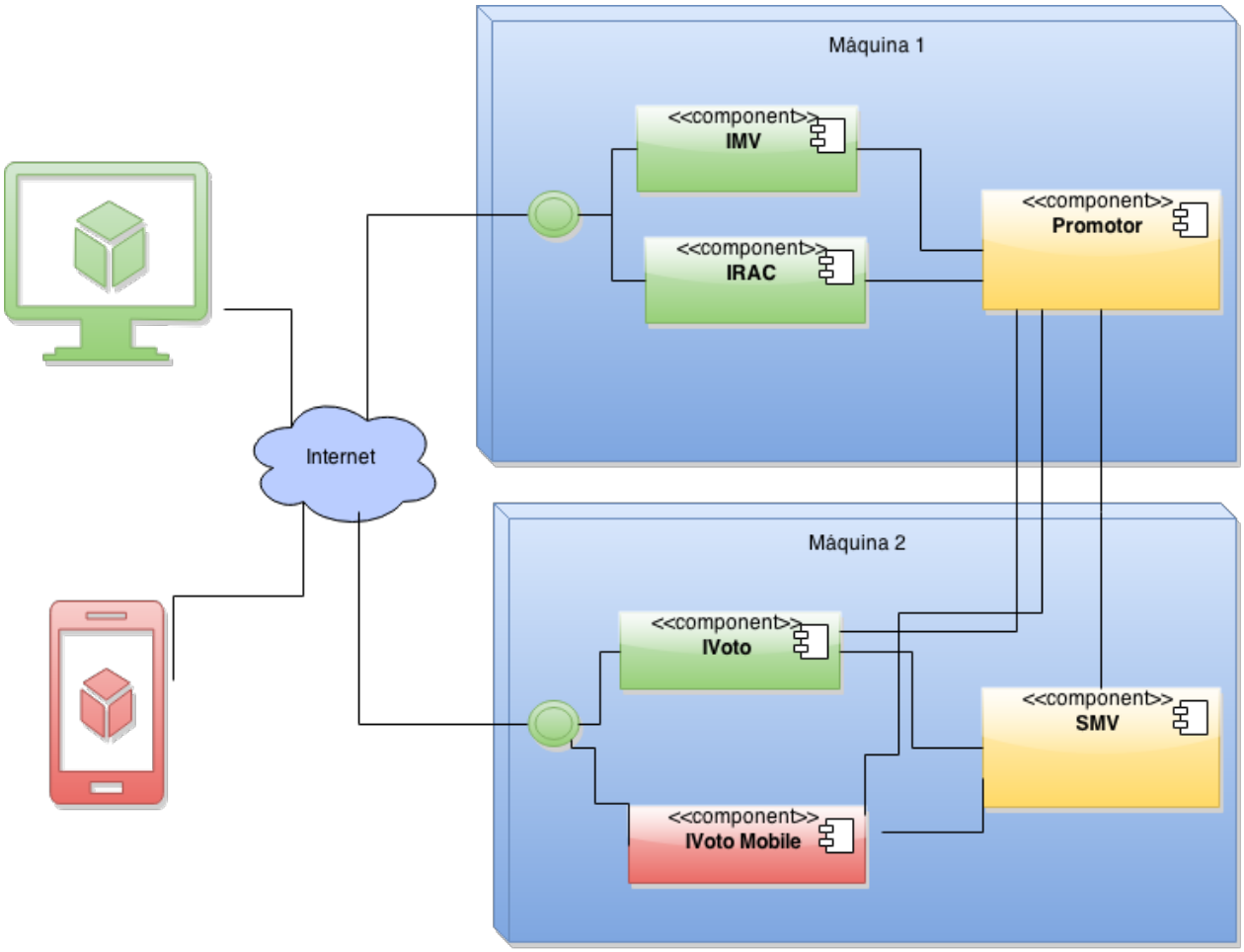


Figura 3.2: Visão geral do sistema CertVote incluindo a solução apresentada

Na Figura 3.2 estão representados os vários componentes do sistema. Os que têm cor vermelha foram os elaborados nesta dissertação e os que têm cor amarela são os componentes do CertVote com que a aplicação comunica. Os componentes a verde pertencem aos CertVote mas não interagem com a aplicação elaborada. De seguida, serão detalhados os 6 componentes:

- Promotor - Componente responsável pela gestão de todos os dados da eleição. Este componente oferece serviços que possibilitam a validação de credenciais, recolha de boletins, segredo partilhado, entre outros.
- Sistema de Mesa de Voto (SMV) - Componente que faz de urna. No SMV são guardados todos os votos.
- Interface de administração do cliente (IRAC) - Permite ao cliente inicializar e finalizar a votação, assim como permite a visualização de resultados.
- Interface web para mesas de voto (IMV) - Permite recuperação de palavra passe, visualizar e alterar o estado do eleitor.
- Interface de votação eletrónica (IVOTO) - Interface com a qual o eleitor interage. Ela permite realizar o voto.
- Interface de votação eletrónica para *smartphones* (IVOTO-MOBILE) - Interface com a qual o eleitor interage. Ela permite realizar o voto.

Destes componentes, o Promotor e o SMV são os componentes "core"que o IVOTO-MOBILE desenvolvida comunica. Os outros 3 componentes (IRAC , IMV e IVOTO) são interfaces web, que pertencem ao sistema CertVote. A comunicação entre os vários componentes é efetuada por web services REST com mensagens em formato JSON.

Apesar de independentes, os componentes estão agrupados por duas máquinas, a do Promotor (contém IRAC, IMV e Promotor) e a do SMV (contém IVOTO, IVOTO-MOBILE e SMV). Este agrupamento foi feito para segregação de funções e poderá ser revisto se necessário, pois todos os componentes são independentes. Em relação à base de dados, apenas o Promotor e o SMV têm acesso, sendo esta localizada fora das respetivas máquinas do Promotor e SMV. O gestor de base de dados utilizada é Postgres.

No sistema de produção da Multicert, o tráfego que é recebido nas máquinas passa primeiro por um *proxy* que verifica a integridade e o conteúdo dos pacotes por questões de segurança e passa por um *load balancer* que reencaminha o pacote para o servidor com menor carga. O IVOTO-MOBILE foi concebido com o intuito de ser escalável, e por isso, caso seja necessário, permite correr várias instâncias do componente.

O acesso ao sistema de voto do CertVote é feito através de web browser, sendo apenas possível aceder externamente às aplicações web IMV, IRAC, IVOTO e IVOTO-MOBILE. O Promotor e o SMV estão acessíveis apenas internamente.

Para concluir, o IVOTO-MOBILE foi o componente criado para permitir comunicação com o sistema de voto através de um *smartphne*.

3.2.2 Navegação na aplicação

Na aplicação proposta, o processo de votação é constituído por 4 fases: identificação da votação (U1), autenticação do votante (U2), realização do votação (U3) e submissão do voto (U4). Os ecrãs e as respetivas siglas estão exemplificadas na Figura 3.3 assim como a navegação possível entre os vários ecrãs.

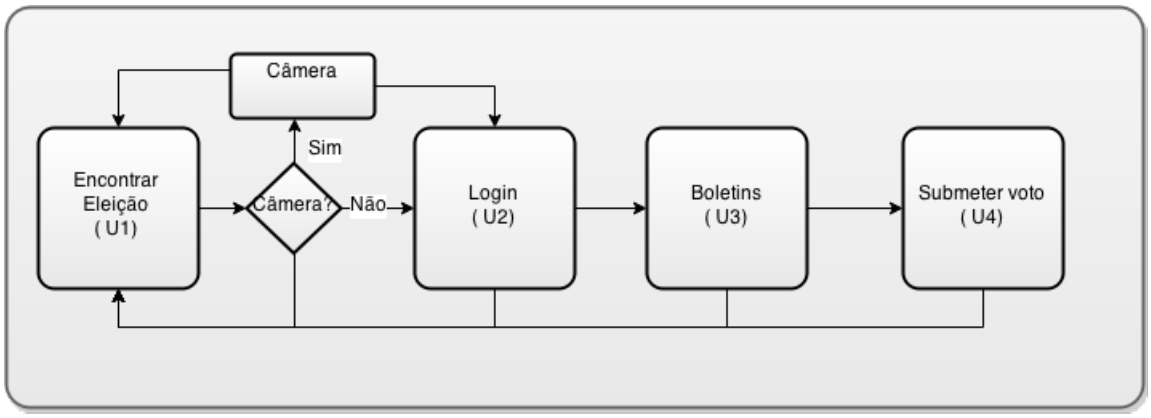


Figura 3.3: Navegação na aplicação

Se o eleitor possuir o QR-CODE da eleição em que pretende votar, poderá apontar a câmara para o código, simplificando a fase de identificação da eleição. Caso não o tenha, poderá sempre introduzir o código manualmente.

Após a identificação da eleição, será apresentada uma lista de campos que serão necessários preencher para autenticar o utilizador. Caso o utilizador introduza credenciais válidas, será levado para um ecrã onde terá vários boletins de voto e um ecrã de resultados. O utilizador poderá escolher uma, várias ou nenhuma opção de voto. Após passar por todos os boletins, serão apresentadas as várias opções selecionadas e um botão para submeter o voto. Após a submissão, a aplicação volta para o ecrã inicial.

3.2.3 Arquitetura física

A arquitetura física apresenta os dois dispositivos móveis (Android e IOS) para os quais foi desenvolvida uma aplicação, apresenta o servidor criado e apresenta o sistema CertVote.

As aplicações móveis comunicam com o IVOTO Mobile através de HTTPS para garantir que a comunicação é cifrada, de ponta a ponta, e assim, um atacante não consegue ler as mensagens interceptadas. Por outro lado, o IVOTO-Mobile e o CertVote comunicam através de HTTP. Este tráfego não é cifrado, porque é considerado que o meio de comunicação entre eles é seguro e é interno, isto é, não utilizam a Internet.

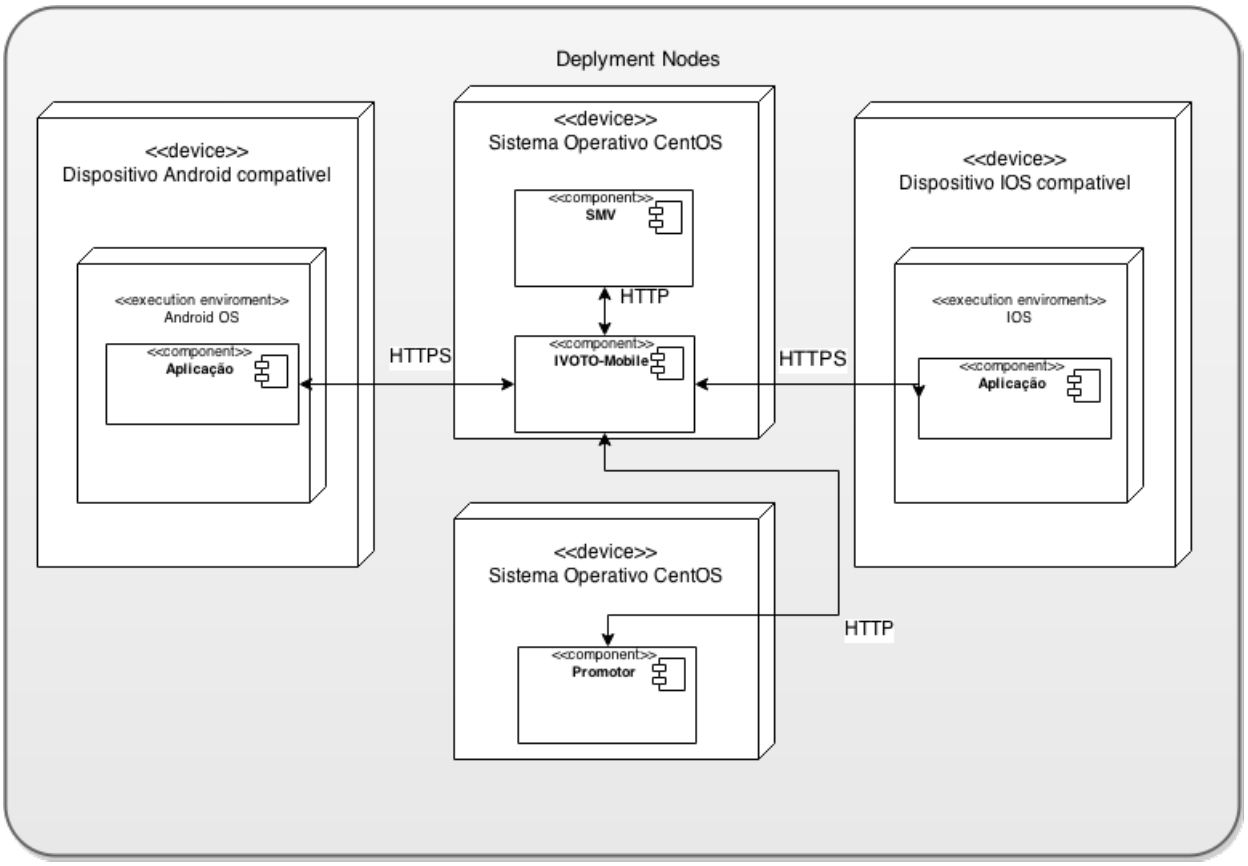


Figura 3.4: Arquitetura física

3.2.4 Arquitetura do modelo criptográfico

Este capítulo visa explanar as interações no sistema que utilizam criptografia. O modelo da Figura 3.5 mostra a interação de um votante com os componentes Promotor e SMV, no ato de autenticação, de votação e anuncia os protocolos criptográficos associados.

Quando se carrega no botão de submeter credenciais, é gerado um par de chaves RSA com tamanho 2048, a sua chave privada é usada para assinar as credencias e a sua chave publica é enviada para o Promotor. Se as credenciais forem aprovadas pelo Promotor, é enviado ao votante um token assinado pela chave pública do Promotor.

Este token permite votar numa eleição, sem revelar a identidade do eleitor porque no token, a identidade do utilizador é substituída por um pseudónimo. De seguida, no ato de votação, é gerada uma chave simétrica 3-DES e o voto é cifrado com ela e depois cifrada com a chave pública associada a essa eleição.

Quando é carregado o botão de submeter credenciais, é gerado um par de chaves RSA com tamanho 2048 e, a sua chave privada é usada para assinar as credencias e a sua chave publica é enviada para o Promotor. Se as credenciais forem aprovadas pelo Promotor, é enviado ao votante um token assinado pela chave pública do Promotor.

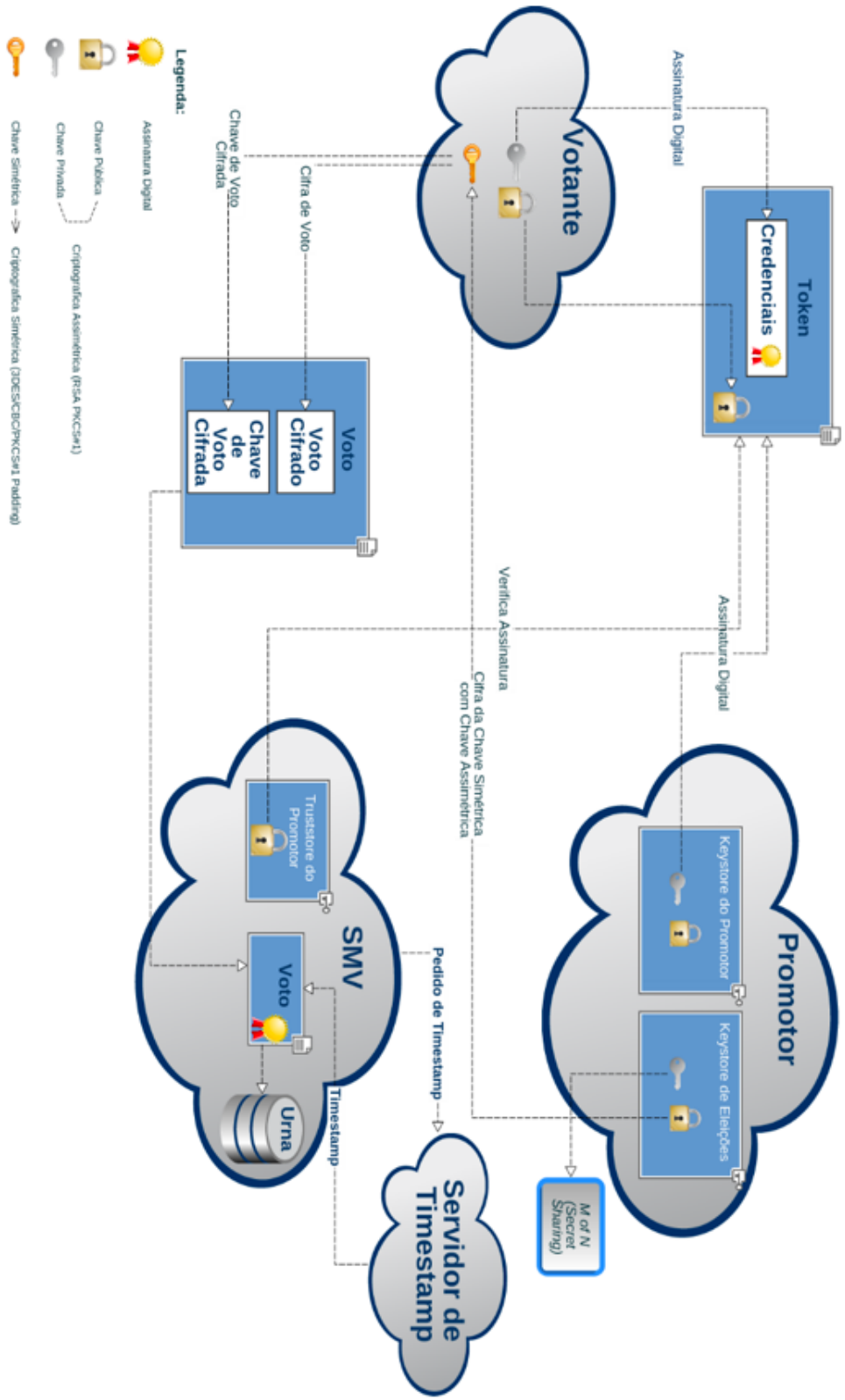


Figura 3.5: Modelo criptográfico CertVote

Este *token* permite votar numa eleição, sem revelar a identidade do eleitor, porque nele a identidade do utilizador é substituída por um pseudónimo. De seguida, no ato de votação, é gerada uma chave simétrica 3-DES e o voto é cifrado com a mesma que, é depois, cifrada com a chave pública associada a essa eleição.

São enviados ao SMV o voto cifrado e assinado e um *token*. A assinatura do *token* é verificada pela chave na *truststore* do SMV; se for válida, é pedida ao servidor de *timestamp* uma assinatura temporal, que será adicionada ao voto e posteriormente guardada na base de dados do SMV, contento desta forma a hora exata da submissão. No próximo capítulo será detalhado com mais pormenor os algoritmos de cifra e assinatura utilizados.

A chave pública associada à eleição é criada através de segredo partilhado, um algoritmo que divide a chave privada em um número definido de pedaços, que são entregues aos supervisores da eleição. Para decifrar com a chave privada da eleição é necessário juntar um número igual ou inferior ao número total de pedaços. Para adulterar este sistema é necessário (embora não suficiente) que o número de supervisores corruptos, a trabalharem em conjunto, sejam o número necessário de pedaços definidos no segredo partilhado.

Este modelo garante o anonimato do voto, se for garantido que, o SMV e o Promotor são fidedignos e não partilham informação relevante com nenhuma parte envolvida no processo e que o segredo partilhado não seja quebrado.

3.2.5 Protocolo de comunicação

Existem duas abordagens para arquitetar um serviço web que são: o REST e o SOAP. A abordagem escolhida foi a REST, devido a ser a implementação com melhor desempenho, mais escalável, com mensagens mais curtas e com um tempo de resposta mais baixo[HSA10]. O protocolo de comunicação foi dividido em duas partes, consoante os componentes que interagem.

Aplicação-servidor

```
1 @POST
2 @Path("/{electionCode}/")
3 public MensagemElection electionRequest(@PathParam("electionCode") String
      electionCode);
4 @POST
5 @Path("/{electionCode}/login")
6 public MensagemLogin loginRequest(@PathParam("electionCode") String electionCode,
      @RequestBody  String requestJSON);
7 @POST
8 @Path("/{electionCode}/vote")
9 public Mensagem voteRequest(@PathParam("electionCode")String electionCode ,
      @RequestBody String requestJSON);
```

Código 3.1: Servicos do IVOTO-MOBILE

O protocolo de comunicação entre a aplicação e o servidor utiliza chamadas a servidor *REST*. O servidor consome e envia conteúdo em JSON. No código 3.1 são mostrados os serviços do IVOTO-MOBILE

A resposta destes serviços é uma mensagem em JSON. Esta mensagem é divida em duas partes. Uma que contém uma mensagem com um estado (usado pela aplicação para saber o estado do pedido, se é de erro ou sucesso) e contém outra mensagem com o conteúdo especifico do serviço. Esta divisão foi feita para que, caso o pedido não seja feito corretamente, o servidor não envie informação sobre os campos da mensagem.

```
{
  "mensagem":"A eleicao nao existe",
  "staus":"OK",
  "template":null,
  "promotorKey":null
  ,"image":null,
  "electionName":null,
  "electionTitle":null,
  "electionDescription":null,
  "credentialNames":null,
  "electionCode":null,
  "election":null

}
```

Código 3.2: Mensagem não dividida

A mensagem mostrada no código 3.2 é uma mensagem sem a divisão. Como podemos ver, ela fornece informação que não deve estar disponível para utilizadores que não introduzam dados corretos. A abordagem para resolver este problema foi a divisão da mensagem. Como podemos ver no Código 3.3, caso o pedido não seja bem formulado ou incorreto, não é enviada informação critica na resposta. Com esta abordagem, é mais trabalhoso a um atacante descobrir como pode subverter o sistema.

```
{"mensagem":{
  "mensagem":"A eleicao aberta",
  "status":"OK"},
"electionResponse":null}
```

Código 3.3: Mensagem dividida

De seguida será detalhado o conteúdo das mensagem e a sequência das ações na comunicação entre o servidor e a aplicação.

```
1
2  @GET
3  @Path("/{electionCode}/")
4  public MensagemElection electionRequest(@PathParam("electionCode") String
       electionCode);
5
6  @POST
7  @Path("/{electionCode}/login")
8  public MensagemLogin loginRequest(@PathParam("electionCode") String electionCode,
       @RequestBody  String requestJSON);
9
10 @POST
11 @Path("/{electionCode}/vote")
12 public Mensagem voteRequest(@PathParam("electionCode")String electionCode ,
       @RequestBody String requestJSON);
```

Código 3.4: Serviços do IVotoMobile

- Serviço S01: ElectionRequest - Faz um pedido ao servidor por uma eleição e recebe uma eleição ou mensagem de erro. Este pedido é feito por HTTPS. Na tabela 3.5 é detalhado como funciona o serviço.

Tabela 3.8: Serviço S01

Identificador	S01
Nome	ElectionRequest
Descrição sumária	Responsável por verificar o estado de uma eleição, e caso ela esteja aberta, devolver as credenciais
Utilizado em	U1
Informação de entrada	Código da eleição
Informação de saída	Caso a eleição esteja aberta, é enviada para a aplicação uma mensagem com as credenciais, chave pública da eleição entre outras informações relevantes. Caso a eleição não esteja aberta, é enviada uma resposta apropriada ao estado da eleição (ainda não abriu, eleição fechada ou não existente)

O diagrama de sequência da Figura 3.6 representa a comunicação da aplicação móvel com o Servidor e vice-versa. O código das respostas encontra-se no apêndice 2 B.

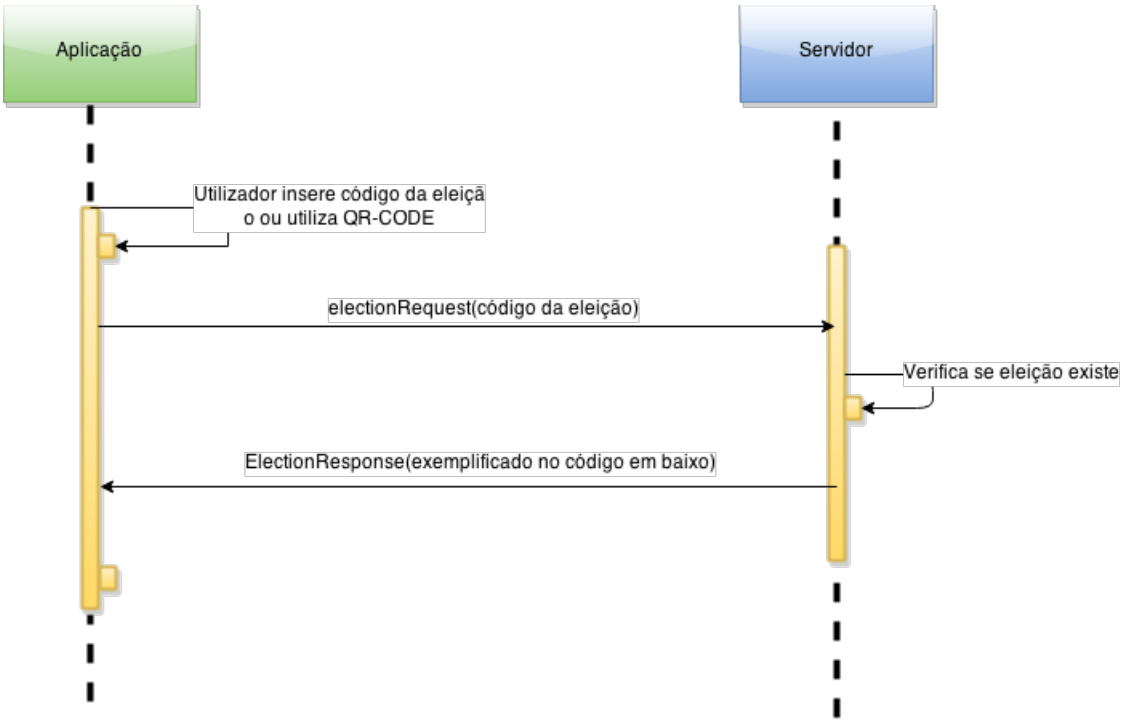


Figura 3.6: Diagrama de sequência entre o S01 e a aplicação

- Serviço S02: LoginRequest - Envia credenciais, recebe um token e os boletins de voto caso, esteja correta. Este pedido é feito por HTTPS. Na tabela 3.6 é detalhado como funciona o serviço.

Tabela 3.9: Serviço S02

Identificador	S02
Nome	MensagemLogin
Descrição sumária	Responsável por verificar se as credencias estão corretas
Utilizado em	U2
Informação de entrada	Credenciais, credenciais assinadas, código da eleição e chave pública
Informação de saída	Caso as credenciais estejam corretas, é enviado um *token* e os boletins

O diagrama de sequência da Figura 3.7 representa a comunicação da aplicação móvel com o Servidor e vice-versa. O código das respostas encontra-se no apêndice 2B.

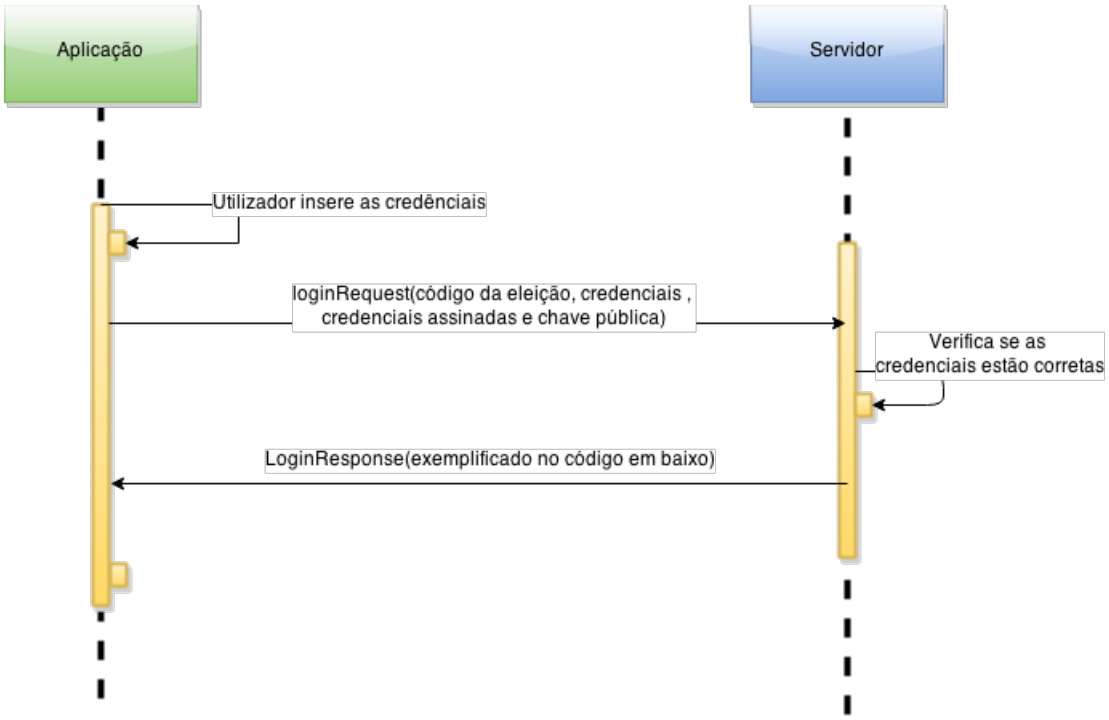


Figura 3.7: Diagrama de sequência entre o S02 e a aplicação

- Serviço S03: VoteRequest - Envia um voto e um token e o servidor responde se o voto foi bem submetido ou não. Este pedido é feito por HTTPS. Na tabela 3.7 é detalhado como funciona o serviço.

Tabela 3.10: Serviço S03

Identificador	S03
Nome	VoteRequest
Descrição sumária	Responsável por verificar a validade do token e do voto e submeter o voto.
Utilizado em	U4
Informação de entrada	Lista de boletins, boletim assinado e chave simétrica encriptada
Informação de saída	Mensagem de sucesso se o voto for bem submetido ou de erro caso não o seja.

O diagrama de sequência da Figura 3.8 representa a comunicação da aplicação móvel com o Servidor e vice-versa. O código das respostas encontra-se no apêndice 2 B.

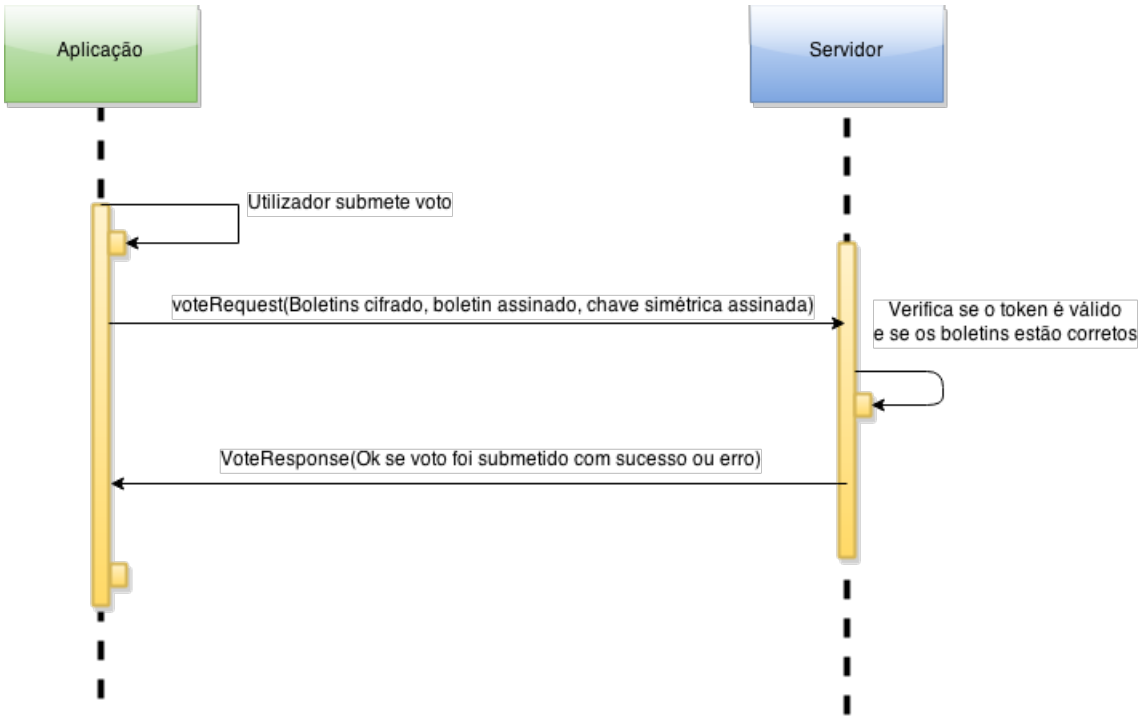


Figura 3.8: Diagrama de sequência entre o S03 e a aplicação

Servidor-CertVote

De acordo com o contrato de confidencialidade assinado entre mim e a Multicert, não me é permitido a divulgação da comunicação interna entre os serviços do projeto CertVote. No entanto, na secção 4.1.2 é explicado, em alto nível, como funciona a comunicação.

3.3 Segurança

Para estudar a segurança de um sistema temos primeiro de definir duas peças muito importantes que são o que pretendemos proteger e quem nos quer atacar. Nesta análise é utilizada a lei de *Murphy*, que enuncia "tudo o que pode correr mal, vai correr mal", permitindo assim identificar os piores casos possíveis. Com este conhecimento, é possível desenhar soluções para esses casos. É importante referir que ao longo desta analise, todo o sistema CertVote é avaliado e não apenas a solução elaborada.

3.3.1 O que se pretende proteger

As eleições têm o requisito de ser anónimas e o seu resultado correto. Para alcançar o estes requisitos é necessário analisar o que se pretende proteger. De seguida são expostos os dados que pretendemos proteger no sistema.

1. As credenciais.
2. O *token*.
3. O voto.
4. A aplicação móvel.
5. O servidor.
6. A comunicação de ponta-a-ponta.

3.3.2 Atacante

Existem várias pessoas com interesse em atacar uma eleição. Podemos pensar que os atacantes podem estar diretamente ligadas à eleição como: os candidatos, os partidos, os eleitores, os supervisores ou são pessoas externas à eleição. Por isso, os atacantes podem ter propósitos diferentes ao atacar o sistema.

3.3.3 Pontos de ataque

Os pontos de ataque são zonas onde o sistema pode ser atacado. Na figura 3.9, podemos ver que o sistema pode ser atacado de várias maneiras diferentes.

1. O atacante pode usar engenharia social para descobrir as credenciais do eleitor.
2. O atacante pode atacar o sistema operativo do *smartphone* e comprometer a memória do telemóvel.
3. O atacante pode atacar a aplicação fazendo com que a mesma tenha comportamentos não esperados, ou não permita ao eleitor utilizar a aplicação.
4. As comunicações podem ser interceptadas ao longo do caminho.
5. O servidor está sempre exposto à Internet.
6. O atacante pode atacar a máquina do servidor.
7. O atacante pode atacar os serviços que ocorrem no servidor.
8. As máquinas dos servidores podem sofrer ataques físicos.

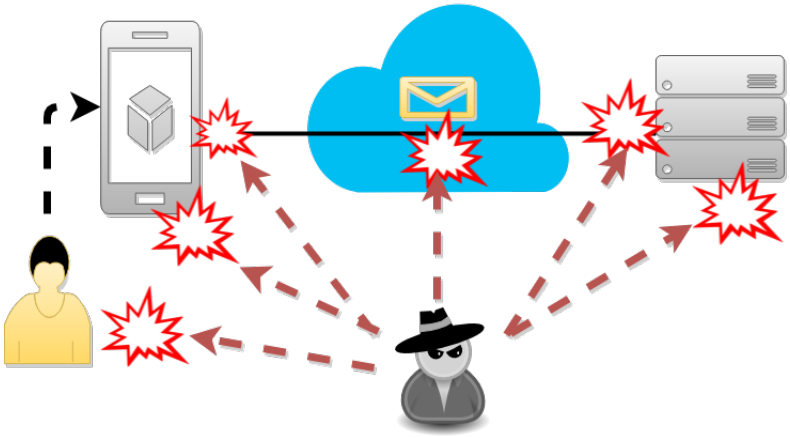


Figura 3.9: Pontos de ataque

3.3.4 Modelos de ataque

Os modelos de ataque são narrativas que devem ser realizados para aprofundar o conhecimento sobre os nossos atacantes. Estes modelos podem ser divididos em 3 categorias: centradas nos atacantes, centradas no software e centradas no conteúdo.

3.3.4.1 Centradas nos atacantes

Para criar um sistema seguro, temos de nos antecipar aos nossos atacantes. De seguida, vai ser exemplificado os objetivos de um atacante e como ele tenciona atacar o sistema, de uma forma imaginária, para tentar precaver o sistema de problemas futuros.

1. **Supervisor** - Os supervisores da eleição podem juntar os seus segredos para tentar decifrar os votos. Para isso, eles teriam de comprometer o SMV. Os objetivos deles passariam por alcançar a urna ou descobrir um voto. Têm um conhecimento fraco de informática. Supõem-se que, no caso de precisarem de conhecimentos informáticos, eles comprem o serviço a BlackHats.

2. **Partido ou candidato** - Tanto os partidos como os candidatos podem ser motivados por dois objetivos diferentes: o primeiro é alterar o resultado das eleições introduzindo votos ou alcançando a urna e o segundo é não permitir votar. Têm um conhecimento muito fraco de informática e segurança. Supõem-se que, no caso de precisarem de conhecimentos informáticos, eles comprem o serviço a BlackHats.

3. **BlackHat** - Os BlackHats atacam por motivos simples como orgulho, reconhecimento, diversão e realização pessoal ou então por outro tipo de motivação como dinheiro e poder. Têm um conhecimento muito elevado de informática e segurança.

4. **Hacktivistas** - A sua motivação é política ou humanitária. Têm um conhecimento muito elevado de informática e segurança.

5. **Serviços secretos e empresas de segurança** - A sua motivação é a possibilidade de manipular e ter poder sobre os vencedores das eleições ou descobrir votos de eleitores. Eles possuem vários *Zero days exploits* que podem comprometer sistemas. Para além disso, os serviços secretos têm acesso às redes de comunicação. Apenas sistemas muito protegidos conseguem resistir a ataques vindos deles. Têm um conhecimento muito elevado de informática e segurança.

Nas figuras seguintes, são ilustrados as várias árvores de ataques que podem ser utilizadas contra o sistema. Na Figura 3.10, é nos apresentada uma possível árvore de ataque por parte de um Supervisor.

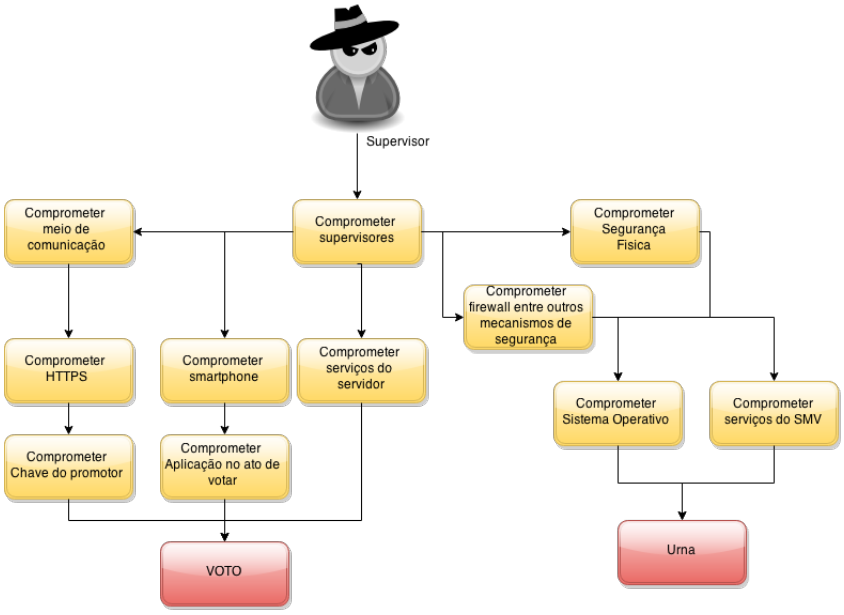


Figura 3.10: Árvore de ataque por um Supervisor

Na Figura 3.11, é nos apresentada uma possível árvore de ataque por parte de um partido ou candidato.

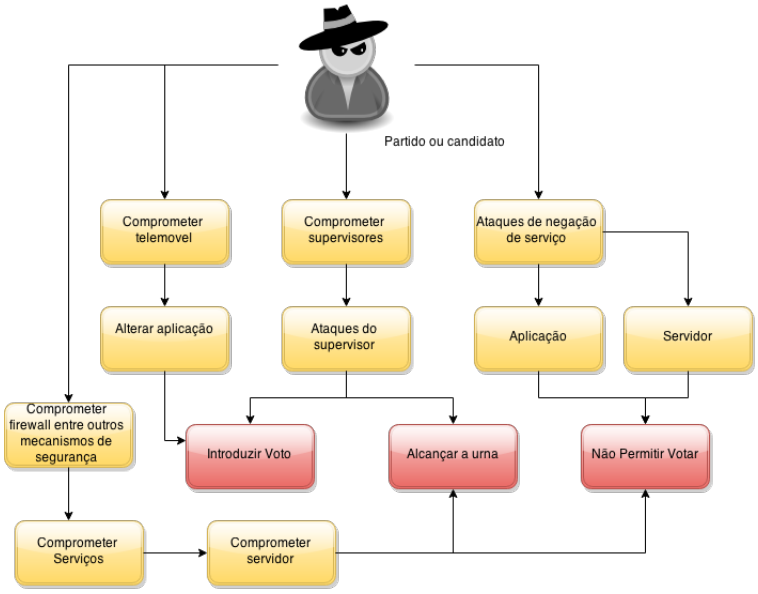


Figura 3.11: Árvore de ataque por um candidato ou partido

Na Figura 3.12, é nos apresentada uma possível árvore de ataque por parte de um BlackHat.

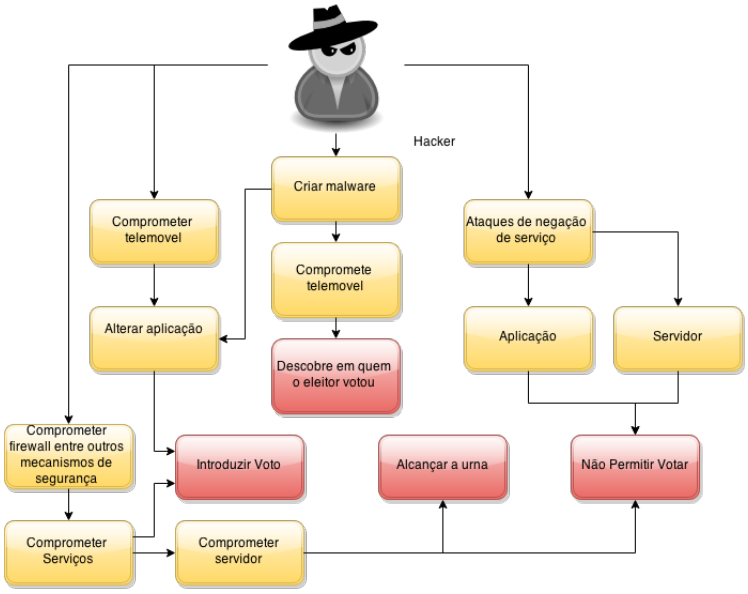


Figura 3.12: Árvore de ataque por um BlackHat

3.3.4.2 Centradas no conteúdo

O sistema foi concebido para proteger toda a comunicação entre o *smartphone* e o servidor, no entanto, é preciso definir quais são as informações mais críticas.

As principais informações necessárias proteger são:

1. **Credenciais do eleitor** - Se for possível a um atacante descobrir as credenciais de um eleitor, ele pode as usar de diversas maneiras, consoante o fato da eleição permitir, ou não, votar mais de uma vez.

2. **Token** - Se for possível a um atacante descobrir o *token* de um eleitor, ele não pode fazer nada se não tiver a chave privada RSA, com a qual o token foi assinado.

3. **Voto** - Se for possível a um atacante conseguir descobrir em quem o eleitor votou, ou interceptar um voto, e descobrir o seu conteúdo, todo o anonimato do voto é perdido. Outro tipo de ataques pode envolver alteração do voto.

		Voto único	Mais de 1
Credenciais	Já votou	-	✓
	Não votou	✓	✓
Token	Já votou	-	-
	Não votou	-	-
Voto	Já votou	✓	✓
	Não votou	✓	✓

Tabela 3.11: Impacto da captura da informação.

A partir da tabela 3.8, podemos ver o impacto, que o fato de um atacante descobrir uma das 3 informações acima descritas, tem no resultado da eleição.

É assumido que as palavras passe são geradas a cada eleição, e por isso, quem souber as credenciais do eleitor, apenas vai poder atacar nessa eleição. Por isso, se a eleição for de voto único e o eleitor já tiver votado, o fato do atacante conhecer as credenciais não tem relevância, pois não altera o resultado da eleição, dado que o sistema deteta que um voto com aquela credencial já foi submetido e não permite votar. Por outro lado, caso o votante não tenha votado, o atacante pode votar por ele e o voto submetido pelo atacante é contado. Assim, o eleitor fica impossibilitado de votar. Caso a eleição permita votar mais de uma vez, e um atacante saiba as credenciais de um eleitor, ele pode votar e alterar o voto do mesmo; no entanto, com este sistema, caso um eleitor seja coagido a votar pode alterar o voto após a coação, se a eleição ainda estiver aberta. Isto impossibilita coação em larga escala.

No caso do *token*, ele tem um tempo de vida curto e se for capturado (e assumindo que tem a chave privada que o assinou), só pode ser utilizado uma vez. Se o utilizador já usou esse token não há nada que o atacante possa fazer com ele, no entanto, caso o atacante consiga capturar o *token* e votar antes do eleitor, o voto do atacante era contabilizado e não era permitido ao eleitor votar.

Caso a eleição seja uma que se pode votar mais que uma vez e o atacante votasse antes do eleitor, ele receberia uma notificação da aplicação a dizer que o seu voto não foi submetido com sucesso e que deveria repetir o ato de votar. Assim, se o eleitor voltar a votar, o fato do atacante saber o token não afeta o resultado da eleição.

É assumido que o voto capturado está decifrado, desta forma, o atacante pode divulgar o voto do eleitor, acabando com o anonimato.

3.3.4.3 Centradas no software

O componente servidor e o móvel vão ser analisadas em separado, pois são dois componentes são totalmente diferentes. Cada um pode ser atacado de maneira diferente.

Ameaças ao servidor

1. **Falhas no sistema operativo** - Falhas no sistema operativo podem permitir que um atacante aceder à maquina onde corre.
2. **Negação de serviço** - Os atacantes podem, através de negação serviço, fazer com que o servidor fique *offline* por não ter capacidade de resposta aos pedidos.
3. **Erros no software de terceiros** - Erros na aplicação de terceiros podem abrir falhas no sistema.

Ameaças à aplicação

1. **Falhas no sistema operativo** - Falhas no sistema operativo podem permitir que um atacante aceda à memória do dispositivo.
2. **Malware** - Software malicioso pode provocar comportamentos anormais na aplicação e recolha de informação protegida.
3. **Erros no software de terceiros** - Erros na aplicação de terceiros podem abrir falhas na aplicação.

Capítulo 4

Implementação

Neste capítulo são apresentados os detalhes mais específicos da implementação do sistema elaborado. É efetuada também uma análise à criptografia e tecnologias utilizadas.

4.1 Detalhes de implementação

Nesta secção é explicada, em detalhe, a criação do servidor e da aplicação. Várias decisões foram tomadas para cumprir os muitos requisitos propostos.

4.1.1 Criptografia

Durante a realização das aplicações, vários princípios criptográficos foram utilizados. Toda a comunicação, entre a aplicação e o exterior, é feita através de HTTPS, permitindo criar um canal de comunicação seguro num meio inseguro. O protocolo implementado no HTTPS foi o TLS v1.2, por ser o protocolo mais forte e recomendado, segundo a OWASP [Pro14]. Para complementar o HTTPS, as aplicação utilizam *Certificate Pinning*, que será detalhado mais adiante.

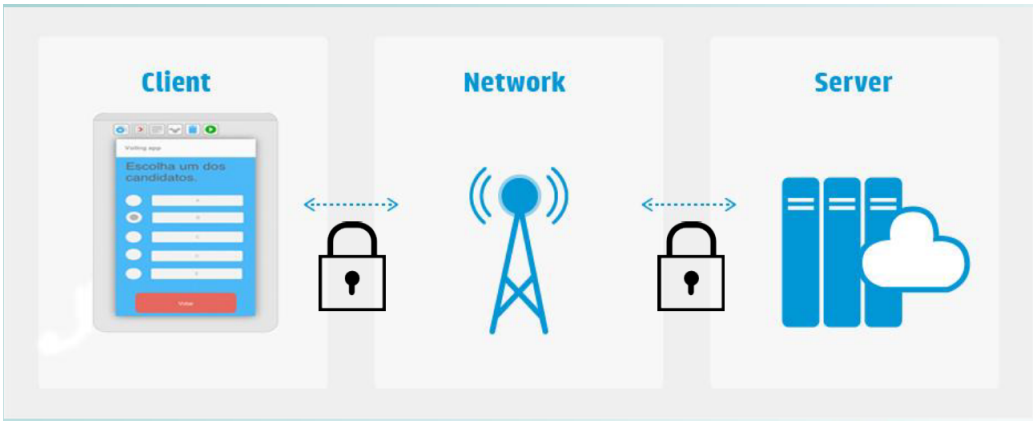


Figura 4.1: Comunicação com segurança em vários pontos da comunicação

No processo de autenticação é gerado um par de chaves RSA de 1024 bits. A chave privada RSA é usada para assinar as credenciais, usando ECB e um padding PKCS1 v1.5. A chave pública, por outro lado, é enviada para o servidor, a fim de ser utilizada na verificação da assinatura dos dados recebidos.

No processo de envio do voto, este é cifrado com algoritmo 3-DES, que usa CBC com padding PKCS5. É usada chave 3-DES de 168 bits, que é depois cifrada usando a chave pública da eleição. Com isto garantimos que, para ler o voto, é necessário extrair a chave 3-DES com a chave privada da eleição, sendo apenas possível decifrar o voto sabendo os segredos partilhados dos supervisores. Para além disso, os boletins são assinados com chave RSA, gerada anteriormente pela aplicação, garantindo assim que sejam aceites apenas boletins assinados com a mesma chave privada do ato de autenticação.

4.1.2 Servidor

Tecnologias utilizadas

Para elaborar o servidor, algumas tecnologias tiveram de ser dominadas e implementadas. Na Tabela 4.1 são mostradas as várias tecnologias usadas na elaboração do Servidor.

Tabela 4.1: Tecnologias usadas pelo Servidor

JAVA 8	Linguagem utilizada para a construção da aplicação Android e do servidor
Apache Tomcat 8	Versão mais recente do servidor aplicacional Tomcat, onde irá ser executada toda a aplicação
Spring infraestrutura 4	Infraestrutura para a plataforma JAVA, baseada em padrões de inversão de controlo (IoC) e injeção de dependência.
Virtual Box	Programa que permite a virtualização de sistemas operativos. Foi usado para correr o CertVote e o servidor.
Maven	O Maven é um gestor de dependências usado pelo Spring e foi utilizado para injetar as dependências do projeto.

O servidor é a ponte de comunicação entre a Internet e o sistema interno. A sua elaboração foi projetada de forma a que o servidor fosse escalável, modular e seguro.

O uso da infraestrutura Spring ofereceu um modelo e um suporte estrutural para o desenvolvimento. Como permite a instalação em qualquer ambiente (JSE ou JEE), é possível executá-lo nos sistemas operativos mais utilizados, cumprindo assim com o requisito *RNF 3* (Sec. 3.1.2). O Spring usa injeção de dependências, fazendo com que seja possível associar uma interface a uma dependência sem necessidade de código de instanciação. Através de injeção de dependências, foi possível reutilizar os módulos usados no sistema CertVote. Com esta abordagem foi possível reutilizar a segurança e a qualidade dos módulos do CertVote, aumentando assim, a segurança do servidor.

Os módulos dos quais o servidor depende são:

1. Um gestor de *keystores* que fornece a chave pública da eleição.
2. Um gestor de *tokens*, que verifica a integridade de um *token*.
3. Cache de boletins, onde estão armazenados os boletins.
4. Um sistema de auditoria, que guarda os registos.
5. Ferramenta de verificação de assinatura.

Detalhes da implementação

Na Figura 4.2 é mostrado o que acontece a cada pedido realizado ao serviço 1 (Sec 3.2.5), os módulos que utiliza, assim como, as suas funções. Este serviço não faz pedidos externos, porque utiliza uma cache de boletins e eleições. Neste momento, a cache não é partilhada com o servidor do I-Voto, mas a sua implementação já esta contemplada na arquitetura.

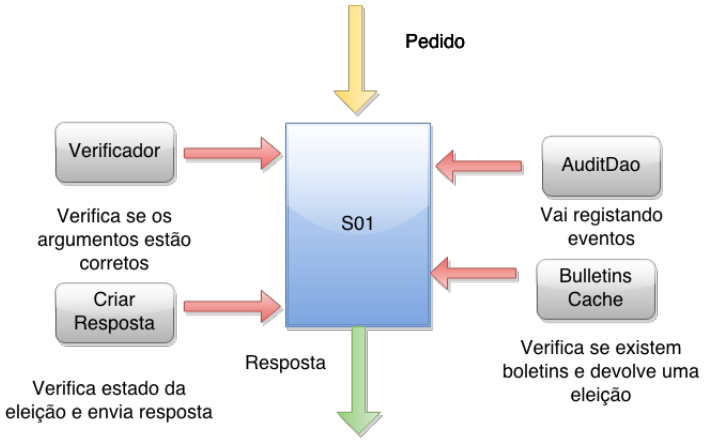


Figura 4.2: Componentes do Serviço 1

Na Figura 4.3 é exemplificado o que acontece a cada pedido feito ao serviço 2, os módulos que utiliza, assim como, as suas funções. Este serviço faz um pedido HTTP ao Promotor, que verifica as credenciais e recebe um token, em caso de sucesso.

Figura 4.3: Componentes do Serviço 2

Na Figura 4.4 é ilustrado o que acontece a cada pedido realizado ao serviço 3, os módulos que utiliza, assim como, as suas funções. Este serviço faz um pedido HTTP ao SMV para submeter o voto, do qual recebe uma resposta a confirmar se o voto foi submetido com sucesso, ou não.

Figura 4.4: Componentes do Serviço 3

4.1.3 Aplicação

Aplicação nativa ou híbrida

Para realizar as aplicações poder-se-ia ter optado criar uma aplicação nativa para cada um dos sistemas operativos, IOS e Android, ou usar uma ferramenta que permita compilador um código para os vários sistemas operativos. Como a segurança é o foco principal desta aplicação, as principais ferramentas multiplataformas (PhoneGap, Titanium e Xamarin) foram estudadas, chegando-se, no fim, à conclusão que as aplicação nativas são mais seguras[Pac13]. As premissas que levaram a esta conclusão foram:

1. O PhoneGap e o Titanium possuem configurações de SSL inseguras e não têm encriptação de conteúdo.
2. Ambas as plataformas têm validações de certificados inseguras.
3. O PhoneGap e o Titanium pedem permissões excessivas no Android.
4. O Titanium não tem restrições à comunicação entre domínios cruzados.
5. Ambas as plataformas utilizam componentes inseguros de terceiros.

Cuidados especiais em aplicações móveis

As aplicações móveis são um tipo de software que requer cuidados diferentes das aplicações normais porque, para criar uma aplicação segura, os programadores tem de estar consciente, dos seguintes fatos:

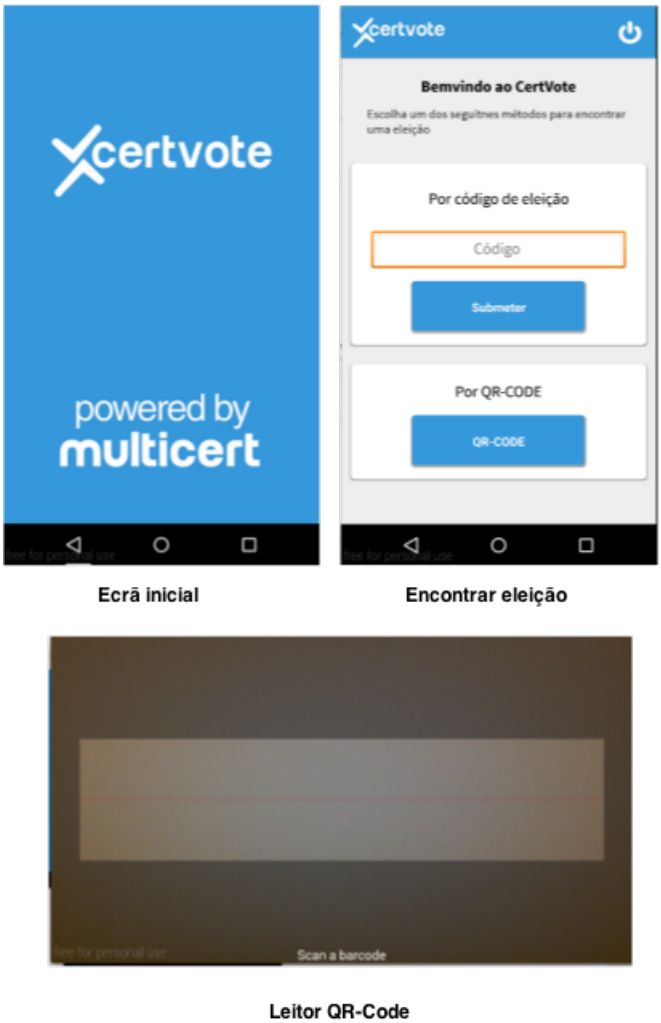
1. A maioria dos erros advêm dos programadores se preocuparem em implementar funcionalidades, sem pensar na segurança.
2. As aplicações devem ser modeladas ao sistema operativo onde correm.
3. Os *smartphones* passam a maioria do tempo ligados à Internet.
4. Os utilizadores não usam os dispositivos com a segurança em mente
5. Os utilizadores caem facilmente em ataques de engenharia social.

4.1.3.1 Android

Nesta seção, são apresentados os ecrãs finais da aplicação, as tecnologias usadas, as boas práticas seguidas e alguns detalhes da implementação.

Ecrãs finais

A Figura 4.5 e 4.6 são os resultados finais da aplicação Android. A aplicação foi desenhada para ser responsiva.



Ecrã inicial

Encontrar eleição

Leitor QR-Code

Figura 4.5: Resultado final Android

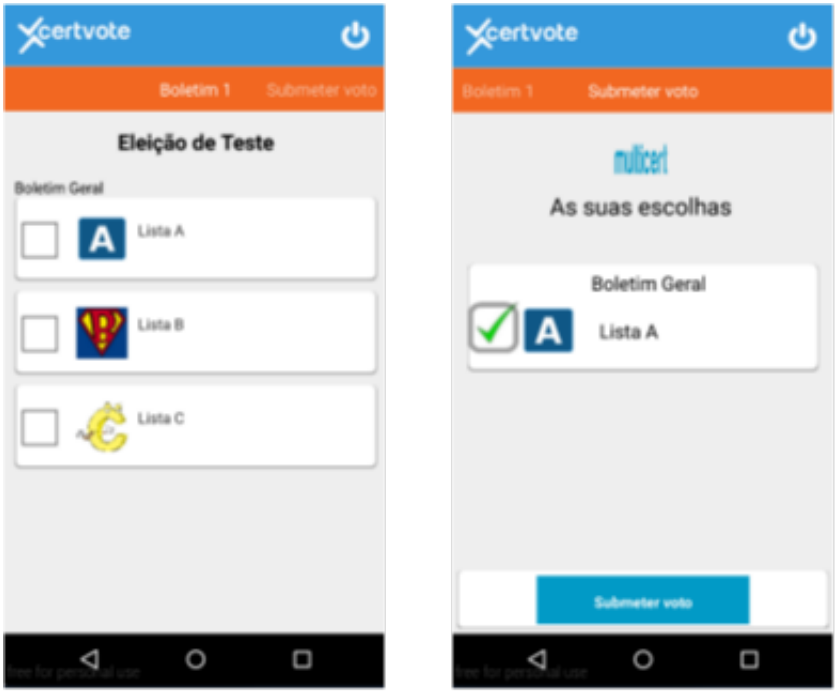


Figura 4.6: Resultado final Android

Tecnologias utilizadas

Para elaborar a aplicação Android, várias tecnologias tiveram de ser dominadas e implementadas. Na Tabela 4.2 são mostrados as várias tecnologias usadas na elaboração da aplicação Android.

Tabela 4.2: Tecnologias usadas para a aplicação Android

Java	Linguagem de programação para Android
Android Studio	IDE de desenvolvimento Android.
ZXing	Permite manipular JSONs.
Robotium	Plataforma para testes funcionais.
SSLPinning	Permitiu implementar o SSL Pinning.
Jackson	Permite manipular JSONs.
Dex2jar	Permite converter DEX para JARs.
JDGui	Permite descompilar JARs.

Boas práticas

A implementação da aplicação Android utilizou vários guias de boas práticas. No sítio do Android[And14] há um conjunto de boas práticas, que foram seguidas tendo em vista o funcionamento correto e seguro da aplicação. Na tabela 4.3 apresenta-se uma lista gerada a partir desse guia:

Tabela 4.3: Lista de Boas Práticas do Android

Título	Descrição	Completo
Guardar dados	A aplicação foi construída de modo a que nenhum conteúdo fosse escrito em disco, evitando assim, a maior ameaça às aplicações Android que é a leitura abusiva.	✓
Uso de *Content providers*	A comunicação é um ponto de ataque e, por isso, a aplicação foi construída de modo a não interagir com outras aplicações do *smartphone*. Para alcançar o objetivo foi preciso embutir o leitor de QR-CODE na aplicação e foi necessário colocar, no *AndroidManifest* da aplicação, o código "android:exported=false"para evitar a interação com outras aplicações	✓
Uso de permissões	O guia recomenda que se deve usar o mínimo de permissões possível e, por isso, a aplicação apenas usa 3. As permissões são de acesso à câmara, de verificador do estado da Internet e de uso da Internet.	✓
Uso da rede	O guia recomenda o uso de HTTPS em vez de HTTP para comunicar e, por isso, todo o tráfego gerado pela aplicação Android é enviado por HTTPS.	✓
Validação de dados de entrada	Segundo o guia, os campos de entrada devem ter uma validação que certifique, o mais possível, que os dados introduzidos são válidos. Por esse motivo, todos os campos de entrada de dados passam por um validador *REGEX*, que certifica se os dados introduzidos são válidos.	✓
Dados do utilizador	É recomendado que os dados do utilizador não sejam acedidos e, como tal, a aplicação não utiliza dados do utilizador disponibilizados pelo sistema operativo. Outro ponto focado nesta secção é a utilização de registos de programador na aplicação. A aplicação foi construída de modo a não serem gerados registos.	✓
Uso de *WebView*	WebViews não foram utilizadas na aplicação. Este componente é muito vulnerável a ataques. [And14]	✓
Credencias do Utilizador	O guia recomenda não guardar credenciais na aplicação o que foi cumprido e, após o utilizador digitar as credenciais, estas são enviadas o mais depressa possível e depois eliminadas e trocadas por um token. A arquitetura da aplicação foi desenvolvida para cumprir esta recomendação. Após a autenticação, apenas são usados tokens.	✓
Uso de Criptografia	Todas as recomendações apresentadas foram utilizadas, foram apenas usadas criptografias estudadas e sólidas. Foi utilizado o HTTPS como recomendado.	✓
Uso de comunicação entre processos	Como anteriormente dito, colocou-se o código "android:exported=false", o que não permite a comunicação com outros processos. O único mecanismo usado foi o *Intent*, ficando assim, invulnerável a aplicações externas.	✓
Carregar código de forma dinâmica	O código usado na aplicação é criado no momento de complicação do *apk*, não sendo usado código externo à aplicação durante a instalação ou compilação.	✓
Utilização do SKD em prol do NDK	Através do NDK, as aplicação Android podem ser construídas usando a linguagem C e C++. No entanto, o seu uso não é recomendado, pois é fácil ao programado cometer erros graves. Para o desenvolvimento da aplicação foi utilizado o SDK.	✓

Problema no Android

O sistema operativo Android tem um grave defeito que afeta aplicações, como a do voto. O Android não permite programaticamente desligar totalmente uma aplicação, pois ela fica sempre no menu de 2º plano, a menos que o utilizador a retire de lá. Isto é indesejável, porque apesar de um utilizador desligar a aplicação e toda a informação ser apagada, é possível ver, no menu de 2º plano, que a aplicação foi corrida, sendo só eliminado o registo se o utilizador o apagar.

A solução encontrada para mitigar o problema foi, sempre que aplicação vai para segundo plano, o ecrã ficar com o *logo* e, quando for reiniciada, voltar para o ecrã U1.

Instantâneos

Por omissão, as aplicações permitem que sejam tirados instantâneos. Isto pode trazer vários problemas de confidencialidade do voto pois, pode ser criado um *Malware* que tire instantâneos enquanto a aplicação esta a correr. Para solucionar este problema foi usado o código 4.1.

```
1
2
3 getWindow().setFlags(WindowManager.LayoutParams.FLAG_SECURE, WindowManager.
      LayoutParams.FLAG_SECURE);
4 getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
```

Código 4.1: Não permitir Instantâneos

ZXing

Foi utilizado o ZXing que, é uma infraestrutura de processamento de imagens de código de barras. Com a ajuda desta infraestrutura, foi possível, implementar um leitor de QR-Code, para satisfazer o requisito *R02.02* e seguir a boa prática da não utilização de aplicação de terceiros.

Jackson

A ferramenta GSON foi, inicialmente usada, como, a ferramenta para a conversão de JSON. No entanto, ao longo da implementação, verificou-se que o GSON, funciona mal com o *InputStream* que, é utilizado na comunicação HTTPS e, por isso, foi depois trocada por a ferramenta Jackson.

SSLPinning

O SSL foi desenhado para dar confidencialidade e integridade dos dados. No entanto, os ataques *Man in the Middle* usados contra conexões SSL podem fazem com que a ligação fique mais lenta ou caia e que o atacante possa descobrir o conteúdo dos dados enviados.

Nos últimos anos, vários ataques foram descobertos contra o SSL [HWC14] onde são feitos ataques à cadeia de confiança. Para resolver alguns dos problemas do HTTPS, foi introduzido

o conceito de *Certificate Pinning*. Este conceito foi proposto pela Google[SEP12], como um novo método para mitigar os riscos do HTTPS e consiste em: limitar Para o uso do *Certificate Pinning* numa aplicação é necessário criar uma *hash* ou PIN de um certificado e garantir que a cada conexão HTTPS, o certificado que nos estamos a ligar tem o PIN ou hash correto. Este sistema implementado juntamente com o TLSv1.2 estando assim, em conformidade com as boa prática[Pro14] de implementação. Foi usada a biblioteca AndroidPinning para implementar o SSLPinning.

Configurações da aplicação

A aplicação foi concebida para permitir funcionar em várias línguas e permite que a aplicação seja adaptável e, para isso, a aplicação foi desenhada para ler as configurações de um ficheiro JSON. Nestas configurações podemos definir vários aspetos como, por exemplo, os textos da aplicação, mudar os algoritmos criptográficos utilizados, mudar a *KeyStores* e mudar o tipo de fonte. As configurações utilizada alterando apenas o JSON.

Keystore Android

Apenas a partir da API 18 do Android, é que o sistema de *KeyStores* foi implementado no Android. O sistema de guardar chaves na *KeyStores* do Android permite que, as chaves criptográficas e informação sensível, seja guardada na *KeyStore*. Como a aplicação suporta *smartphones* Android a partir da API 13, o sistema foi modelado para usar *KeyStores* em dispositivos que a suportem.

Secure Random

O Secure Random é a maneira recomendada pela OWASP entre outras[And14] para implementar a geração de chaves aleatórias e, por isso, foi utilizado para gerar chaves.

4.1.3.2 IOS

Ecrãs finais

A Figura 4.7 e 4.8 são os resultados finais da aplicação IOS.

Ecrã inicial

Ecrã encontrar eleição

Ecrã de credenciais

Figura 4.7: Resultado final IOS

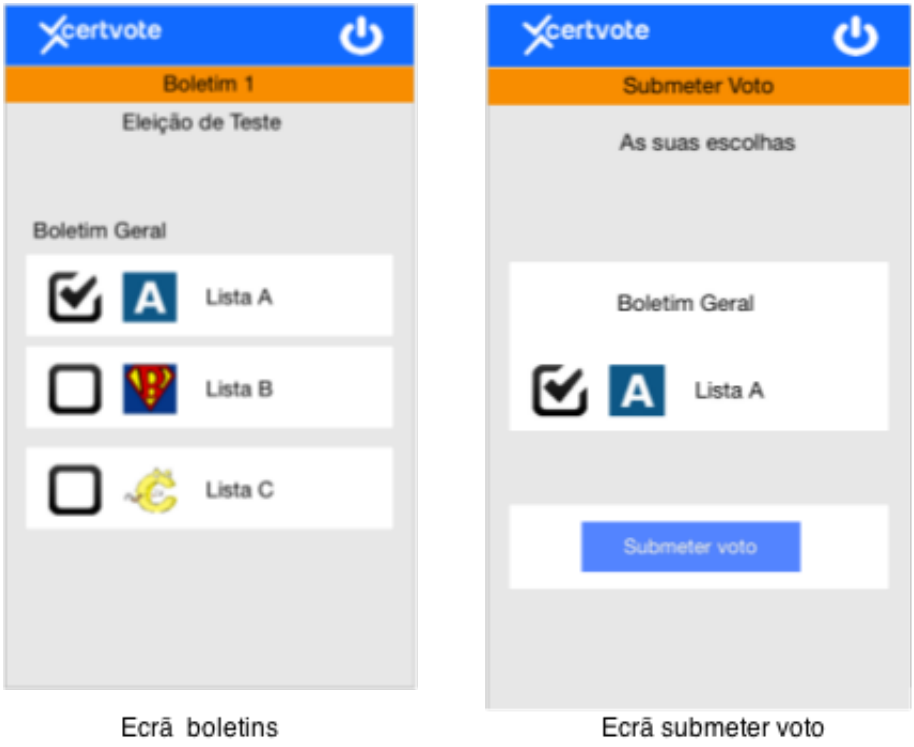


Figura 4.8: Resultado final IOS

Tecnologias utilizadas

Para elaborar a aplicação IOS, várias tecnologias tiveram de ser dominadas e implementadas. Na Tabela 4.4 são mostradas as várias tecnologias usadas na elaboração da aplicação IOS.

Objective X	Linguagem utilizada para a construção da aplicação IOS .
XCode	IDE utilizado para desenvolver código em Objective-C.
RSA Wrapper	Usada para cifrar com o RSA e fazer assinaturas.
SSL-conservatory	Permitiu implementar o *Certificate Pinning* no IOS.

Tabela 4.4: Tecnologias usadas na aplicação IOS

Boas práticas

Para desenvolver a aplicação IOS, foram seguidas várias boas práticas[Inc14], no essencial apresentadas na Tabela 4.5.

Título	Descrição	Completo
Uso de Criptografia	Todas as recomendações apresentadas foram seguidas, sendo apenas usadas criptografias estudadas e sólidas. Foi utilizado o HTTPS com *Certificate Pinning*	X
Evitar BufferOverFlow	No guia é possível ver um conjunto de funções que não devem ser usadas, como o *strcopy* e o *strcat*. Devem ser usadas antes as funções preferencialmente o *strlcopy* e o *strlcat*,	X
Não usar buffer sizes hard-coded	O tamanho do buffer deve estar bem definido.	X
Uso de Privilégios	Deve-se reduzir os privilégios, o mais possível, para limitar os danos causados por um ataque.	X
Validar Input e comunicação entre processos	Campos com erros podem causar problemas, como o Buffer OverFlow, String Attacks e Injection Attacks.	X

Tabela 4.5: Lista de Boas Práticas do IOS

RSA Wrapper

Para fazer uma aplicação IOS, que comunique com um servidor em Java, é necessário fazer algumas operações, a fim dos dados serem compatíveis. Inicialmente, houve alguns problemas no conteúdo das mensagens cifradas com o RSA, pois não era lido corretamente pelo servidor. Com a biblioteca *RSA Wrapper*[ree14], foi possível cifrar com o RSA e comunicar com o servidor Java; no entanto, esta biblioteca *open-source* não continha um módulo de assinatura. Por conseguinte, foi realizado um módulo de assinatura, que foi submetido para a biblioteca *open-source*, através de um *fork*[Gri15].

SSL-Conservatory

Como foi referido na secção do Android, o *Certificate Pinning* é uma metodologia que aumenta a segurança da implementação do TLS e por isso, foi também utilizado no IOS. Foi utilizada a biblioteca *SSL-Conservatory*[iSE13] para fazer o *Certificate Pinning*.

Capítulo 5

Resultados

Nesta secção são mostrados os vários testes unitários e funcionais que foram feitos ao sistema com o intuito de mostrar a qualidade do produto final. Foram estudados os ataques mais conhecidos a aplicações móveis, e feita uma comparação entre a aplicação desenvolvida e a da Estónia.

Por último, são mostrados os resultados das eleições de teste e é feita uma reflexão sobre um questionário realizado.

5.1 Testes de segurança no Android

Como foi bem marcado ao longo da dissertação, a questão da segurança é fundamental. Todo o sistema foi desenvolvido com esse mesmo foco. As aplicações móveis estão na mira dos atacantes e, por isso, é necessário compreender como podem as aplicações estar seguras. Para defendermos a aplicação contra ataques é preciso primeiro definir o que queremos proteger. No caso da solução proposta é o conteúdo que está no ecrã, conteúdo em memória, envio e recepção de mensagens e funcionamento correto da aplicação.

Caso o dispositivo usado para votar tiver *root*, é muito difícil de garantir que a aplicação continua segura, porque um utilizador com *root* pode ter acesso a qualquer conteúdo, sem restrições. Com isto quer-se dizer que, uma aplicação que execute como o utilizador administrador, tem acesso a recursos do sistema, sem ter passar pelo controlo de permissões do sistema. Na implementação da aplicação Android, foi colocado um método que permite desligar a aplicação, caso o dispositivo tenha *root*.

É importante salientar que já foram documentadas diversas falhas nos sistemas operativos móveis. Essas falhas são geralmente corrigidas nas versões superiores dos sistemas operativos. Por isso, a segurança de um smartphone (Android ou IOS) é tanto maior, quanto mais recente for a versão do sistema.

5.1.1 Metodologia utilizada

Não existe uma metodologia exata para o teste de uma aplicação Android, pois as pré-condições são muito flexíveis e a segurança da aplicação depende da versão do sistema operativo, se tem *root*, da rede onde se conecta e de modificações da marca. Por conseguinte, foram utilizadas várias metodologias para testar a aplicação do maior número de maneiras possível.

A primeira metodologia utilizada foi o guia para testes de segurança, para aplicações móveis da OWASP[Pro14], que é divido nas seguintes etapas: recolha da informação, modelo de ameaças e análise de ameaças.

5.1.1.1 Recolha da informação

A recolha de informação foi um processo rápido, pois, a maioria dos dados da aplicação já foi referenciada anteriormente. No entanto, ainda não foi analisado o trafego gerado. Na Figura 5.1 é mostrado esse tráfego, usando HTTP. O conteúdo da mensagem é facilmente capturado por um atacante, se não existir proteção.

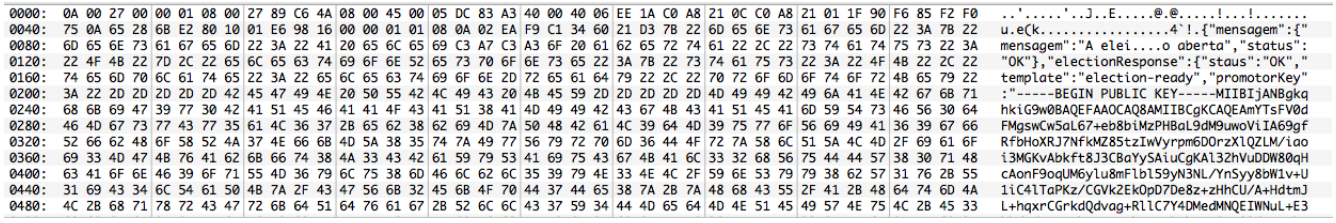


Figura 5.1: Captura de trafego não cifrado

Na Figura 5.2 mostra o tráfego encriptado com *TLS v1.2* e o conteúdo da mensagem está cifrado e ilegível.

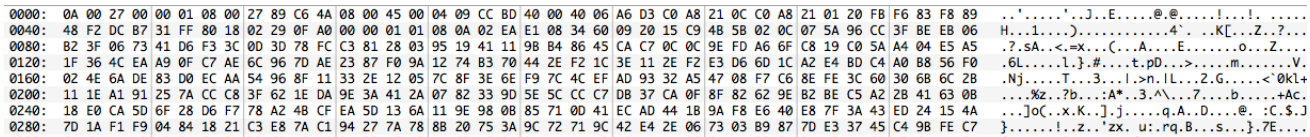


Figura 5.2: Captura de trafego cifrado por TLS v1.2

5.1.1.2 Modelo de ameaças

O modelo de ameaças permite descobrir possíveis ataques ao sistema. O modelo passa por as seguintes etapas: identificar os componentes chave, identificar e dar uma classificação às ameaças a cada um dos componentes e desenvolver proteções contra as mesmas.

Na próxima tabela 5.1, estão descritos as operações realizadas pela aplicação, os possíveis ataques e a proteção. O código por vezes referido está apresentado no Código 5.1 a 5.6.

ID	Ação	Ameaça	Proteção
A1	Utilizador envia conteúdo pela Internet	Um atacante pode interceptar o tráfego e ler o conteúdo dos pacotes	*HTTPS*.
A2	O utilizador carrega no ecrã para introduzir credenciais, fazer escolhas e submeter.	Um atacante pode fazer o ataque *TapJacking* e tirar informação sobre os locais onde o utilizador carrega	Código 5.1
A3	A informação do voto e as credenciais do utilizador aparecem no ecrã	Um atacante pode tirar instantâneos ao ecrã	Código 5.2
A4	Utilizador faz "copiar-colar"das credenciais	Um atacante pode ter acesso ao *clipboard*	Após sair do ecrã de autenticação, o *clipboard* é apagado usando o código 5.3 pois o utilizador já não necessita dessa informação.
A5	Utilizador instala uma aplicação que interfere com os *Intents* da nossa aplicação	Um atacante pode trocar de janela rapidamente e enganar o utilizador. O utilizador pode ser levado a pensar que está na aplicação correta	A implementação do código 5.4 faz com que não seja possível chamar os *Intents* da nossa aplicação.
A6	Utilizador instala uma aplicação que interfere com os *Intents* da nossa aplicação	Um atacante pode trocar de janela rapidamente e enganar o utilizador. O utilizador pode ser levado a pensar que está na aplicação correta	A implementação do código 5.4 faz com que não seja possível chamar os *Intents* da nossa aplicação.
A7	O utilizador perde o *smartphone*, ou é roubado, durante o uso da aplicação	Um atacante poderia desbloquear o telemóvel e votar	A aplicação tem vários contadores, fazendo com que, passados um certo tempo de o utilizador não carregar ecrã, volte ao menu inicial.
A8	O utilizador instala um *malware*	O *malware* do atacante consegue ler os registos	Todos os registos foram apagados da aplicação.
A9	O utilizador tem o modo de*bug* ligado	O atacante tenta usar o *debug* para mexer ou ler conteúdo da aplicação	A aplicação desliga-se se o modo *debug* estiver ligado. O código 5.5 foi usado para detetar o modo *debug*.
A10	O utilizador tem *root* no *smartphone*	Um atacante com acesso ao telefone, poderia ter todo o tipo de acessos	A aplicação desliga-se, se o *smartphone* tiver SuperUser, conseguir informação da *build* ou conseguir executar comandos *sudo*. O código 5.6 verifica as opções anteriores.
A11	O utilizador instala uma replica falsa da aplicação	Um atacante tem acesso ao telefone e instala a aplicação falsa	Verificar hash ao iniciar a aplicação.

Tabela 5.1: Modelo de ameaças

```
"android:filterTouchesWhenObscured="true
```

Código 5.1: Não permitir *TapJacking* de um elemento da interface gráfica

```
"getWindow().setFlags(LayoutParams.FLAG_SECURE, LayoutParams.FLAG_SECURE);"
```

Código 5.2: Não permite instantâneos

```
ClipboardManager clipboard = (ClipboardManager)
getSystemService(Context.CLIPBOARD_SERVICE);
clipboard=null;
```

Código 5.3: Limpa conteúdo do ClipboardManager

```
export="false"
```

Código 5.4: *Intent* não pode ser exportado

```
boolean isDebuggable =  ( 0 != ( getApplicationInfo().flags & ApplicationInfo.
    FLAG_DEBUGGABLE ) );
```

Código 5.5: Verifica se o telemóvel tem o modo *debug* ativo

```
   // Buscar informacao da build
   String tags = android.os.Build.TAGS;
   if (tags != null && tags.contains("test-keys")) {
   exit();
   }

  // verifica se o SuperUser esta instalado
   File file = new File("/system/app/Superuser.apk");
   if (file.exists()) {
     exit();
   }

   // tenta correr comandos em sudo
   Runtime.getRuntime().exec("which su");
```

Código 5.6: Várias abordagens para descobrir se o *smartphone* tem *root*

5.1.1.3 Análise de ameaças

Foi possível proteger a aplicação da maioria das ameaças. No entanto, o ofuscador usado foi o ProGuard e ele não consegue ofuscar texto, como outras ferramentas proprietárias que existem. Se uma aplicação não estiver protegida com ferramentas de ofuscação, nela é muito fácil de fazer engenharia inversa. O utilizador pode ser enganado e levado a instalar uma aplicação que se faz por legítima, roubando-lhe de seguida, as credenciais ou descobrindo em quem o utilizador quer votar.

5.1.1.4 As 10 maiores ameaças

Seguda a OWASP, os seguintes 10 problemas são os mais importantes actualmente em dispositivos móveis; para cada um, mostra-se como foi solucionado ou mitigado.

- M1: Fraco sistema de controlo do servidor - Para mitigar este risco, o código do servidor verifica todos os dados recebidos e a validade dos *tokens*. As configurações do servidor foram escolhidas para aumentar a segurança ao máximo.

- M2: Guardar dados de forma insegura - A aplicação foi concebida para não guardar dados, eliminando, assim, este risco.

- M3: TLS insuficiente - Com o uso de TLS v1.2 e com a verificação do domínio, certificado e introdução do SSLPinning, o risco foi mitigado ao máximo.

- M4: Libertação de dados não autorizada - Foram removidos todos os registos produzidos pela aplicação, introduzido código para prevenir *TapJacking* e o *buffer* do copiar-colar é posto a nulo, no fim da execução. Todos os cenários pensados, com resolução, foram protegidos.

- M5: Pobre autorização e autenticação - A autenticação é sempre feita no lado do servidor e em todas as chamadas, não públicas, é necessário autenticação. O tamanho das palavras passe geradas pelo CertVote tem 6 ou mais caracteres. Por isso, este risco foi eliminado.

- M6: Uso de cifras partidas - A aplicação não guarda a keystore, apenas o PIN para o *Certificate Pinning*. O PIN foi codificado em base 64 para tornar mais difícil a percepção, para um atacante que tente fazer engenharia reversa. A utilização do ProGuard permitiu defender os binários de ataques de engenharia inversa. É utilizado o SHA-1 para assinatura e, devido a natureza da assinatura, esta cifra apesar de não ser recomendada, ainda continua segura.

- M7: Injeção de dados no cliente - Com a não utilização de WebViews e com a aplicação do "export=false"no Manifest, estes ataques foram mitigados ao máximo.

- M8: Decisões de segurança via introdução de dados não confiáveis - Como a aplicação é independente e não comunica com outras aplicações, este risco foi totalmente coberto.

- M9: Sessão mal gerida - Com a destruição e tempo limitado dos *tokens*, a sessão é gerida sempre da forma correta.

- M10: Falta de proteção de binários - Foi aplicado o ProGuard e o PIN foi colocado em base 64. Existem aplicação melhores que o ProGuard para ofuscar código Android, mas são proprietárias.

5.1.2 CertVote Mobile vs VK da Estónia

Há algumas abordagens para o voto através de smartphones, mas não existe nenhuma em que o seu foco tenha sido a segurança. A aplicação **VK** é a aplicação do Governo da Estónia para voto e o seu código fonte está disponível no GitHub. Como foi concebida para ser realmente utilizada e não como protótipo, como a maioria dos trabalhados apresentados no capítulo 2, ela serviu como métrica de comparação com a aplicação desenvolvida.

Na Tabela 5.2, estão descriminados os ataques fazíveis à aplicações de voto, recolhidas no modelo de ameaças (Tabela 5.1), e a proteção do CertVote Mobile e do VK da Estônia.

ID	CertVote	VK
A1	P	P
A2	P	A
A3	P	A
A4	P	A
A5	P	A
A6	P	A
A7	P	P
A8	P	P
A9	P	P
A10	P	A/S
A11	A	A

Tabela 5.2: Comparação: A - Ameaça, P - Protegido, S - Sem dados

Pela análise do quadro anterior, podemos concluir que o CertVote está muito mais protegido contra ameaças, do I-Voting, que o VK.

5.1.3 ViaLab

Figura 5.3: Ihttps://www.nowsecure.com/blog/2014/09/10/introducing-vialab-community-edition/

O ViaLab é uma ferramenta que permite fazer teste de segurança em aplicações Android e IOS. Esta tem um módulo pago e outro gratuito, sendo este último o utilizado neste trabalho. Inclui algumas funcionalidades como captura de pacotes de rede, procura de dados sensíveis, scripts de teste automáticos, entre outras. O ViaLab corre uma enorme variedade de avaliações de vulnerabilidades. No início da avaliação são definidas as palavras-chave a serem analisadas, no caso foram: nome de utilizador, palavra-chave, token e opções de voto. A ferramenta permite que seja também verificado se os parâmetros analisados estão de acordo com as normas: HIPAA, FISMA, ISO 27001, ISO 15408, EFTA, PCI-PA, NERC, SoGP, PCI-DSS, GLB, NIST e COPPA. Concluídos os testes é providenciado um relatório com as descobertas.

O relatório dado a esta aplicação (Apêndice 3**??**) está dividido em três secções: *Setup*, *Network* e *Compliance*. Na primeira secção, *Setup*, o ViaLab encontrou um risco médio, por considerar a complexidade da palavra-passe não ser satisfatória; contudo isto não é correto, porque a verificação da palavra-passe é feita no servidor e apenas admite palavras-passe superiores a 6 caracteres, com números, caracteres especiais e letras maiúsculas e minúsculas, seguindo a norma das 10 maiores ameaças móveis da OWASP (M5: Pobre autorização e autenticação). Excetuando este fato, no relatório é afirmado que a aplicação passa em todos os testes. Ela não conseguiu encontrar dados sensíveis durante o uso da aplicação, efectuou vários ataques como HTTPS *downgrade*, usando SSL *strip*, ataques *men-in-the-middle* e *port-scan*. Assegurou que todo o tráfego corria sobre SSL. Por fim, verificou que todos os dados estavam de acordo com as normas supracitadas.

5.2 Testes unitários e funcionais

Com a finalidade de certificar que a aplicação desenvolvida tem qualidade, foram realizados diversos testes unitários e funcionais, aos componentes mais importantes. Os testes unitários permitem garantir que os resultados dos métodos estão de acordo com as expectativas. Aplicaram-se vários valores de entrada, tendo em vista confirmar o correto funcionamento dos métodos. O JUnit foi a ferramenta utilizada para os testes unitários no Android e no servidor; através dela foram cobertos os principais métodos.

Os testes funcionais permitem avaliar o comportamento da aplicação. As ferramentas utilizadas para testes funcionais foram o Spock, para o servidor e o Robotium, para Android. Com o Spock foi possível testar o comportamento do servidor, incluindo testes de sobrecarga. O Robotium permitiu criar testes de caixa negra à aplicação Android.

5.3 Desempenho

O consumo de memória pela aplicação Android é de 20 MB, no ecrã inicial e no de encontrar eleição (Ver Figura 5.4). No ecrã das credenciais é de 30 MB e no ecrã de resultados varia (mínimo 40 MB). A aplicação consome mais recursos nestes últimos ecrãs, porque há imagens. A aplicação corre fluidamente; no entanto, nos telemóveis mais antigos, tal pode não acontecer. Com algumas otimizações às imagens, este problema é facilmente contornado.

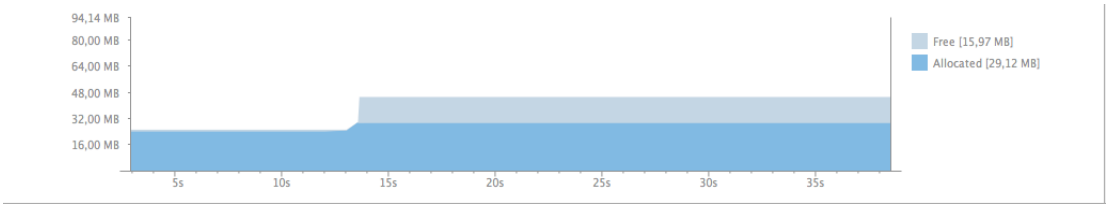


Figura 5.4: Componentes do Serviço 2

5.4 Resultados

Os resultados da aplicação, em termos de exatidão do voto, é de 100 porcento. O sistema criado está preparado para ser integrado no CertVote e ser comercializado.

Foram realizadas eleições de teste usando quer *smartphone*, quer navegador, e ambas as aplicações funcionaram em harmonia.

5.5 Questionário

Foi realizado um questionário com as seguintes perguntas:

1. Dados do inquirido (sexo e idade).

2. Se são, ou não, a favor do voto pela Internet.

3. Meio preferido para votar (Smartphone, PC ou nenhum).

4. Se gostavam de exercer mais vezes o direito de Voto.

Para garantir a integridade dos dados, foi utilizada a plataforma Google Forms, que permite criar inquéritos. Para responder ao inquérito, era necessário ter um conta Google, para confirmar se o inquirido já respondeu ao inquérito, ou não.

Foram 243 inquiridos e, a sua grande maioria pertence a faixa etária dos 18-35. Sendo assim, não é possível extrapolar os dados para a vontade da maioria dos Portugueses, mas permite extrapolar o que faixa etária a baixo dos 35 sobre votação pela Internet.

Os resultados obtidos são apresentados na Figura 5.5 até à 5.7.

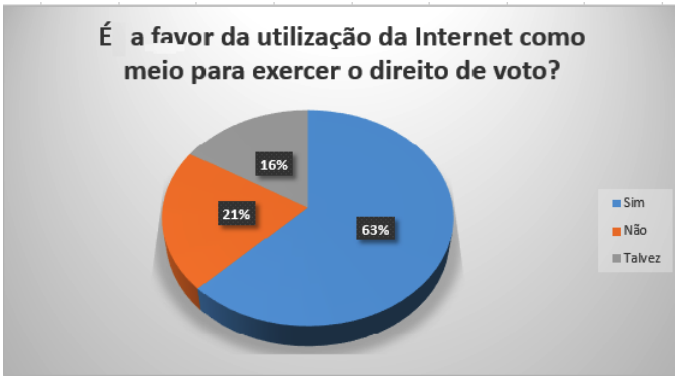


Figura 5.5: Gráfico com a pergunta : "É a favor da utilização da Internet como meio para execer o direito do voto?"

Por este gráfico, podemos concluir que a grande maioria dos inquiridos gostava de votar pela Internet.

Figura 5.6: Gráfico com a pergunta : "Que meio preferia usar para votar pela Internet?"

Deste gráfico, advêm que os utilizadores tanto preferiam votar, por *smartphone*, como por computador.

Figura 5.7: Gráfico com a pergunta : "Gostava de exercer o direito de voto mais vezes?"

Pelos resultados, podemos afirmar que a maioria dos inquiridos, gostava de ser mais vezes chamado às urnas e ter um maior poder de decisão.

Capítulo 6

Conclusões

Neste capítulo são expostas as conclusões desta dissertação.

6.1 Satisfação dos objetivos

O sistema de voto criado foi totalmente integrado no CertVote e já foi usado em eleições de teste. Os requisitos, tanto do servidor, como os da versão Android e IOS, foram satisfeitas. Com os testes de segurança realizados, é possível garantir que estamos perante um sistema seguro e pronto a ser testado em larga escala. Com a análise feita, no capítulo dos testes, é possível afirmar que: a aplicação Android desenvolvida é mais segura que a aplicação utilizada na Estónia. Os problemas da aplicação da Estónia foram identificadas no seguimento deste trabalho autor e soluções foram apresentadas aos autores da aplicação.

6.2 Melhorias para o sistema CertVote

O sistema CertVote foi realizado com o intuito de satisfazer as necessidades dos seus principais clientes, instituições privadas e públicas. Por isso, a sua arquitetura cumpre certas regras que, num cenário eleitoral normal, não teria de cumprir.

Os vários princípios criptográficos aplicados ao voto apresentados no capítulo 2, como a cifra homomórfica, provas sem conhecimento, assinaturas cegas e canais anónimos, podem ser implementados no sistema CertVote. Desta forma, o sistema tem a possibilidade de permitir ao votante verificar se o seu voto foi realmente contado e de se aumentar o anonimato das votações. Com isto, a responsabilidade de verificar a integridade das eleições é também responsabilidade do eleitor, pois pode sempre verificar se o seu voto foi contado corretamente, diminuindo assim, a responsabilidade assumida pelo Promotor e SMV.

Outro aspeto que a plataforma pode melhorar é na escolha de cifras usadas. Segundo a Cisco[Cis14], apesar de ser aceitável usar o *TripleDES* em casos onde o tempo de vida da chave ser curto, poder-se-ia optar pela cifra da próxima geração, como o *AES-GCM*.

No caso da assinatura, é usado *SHA-1*, algoritmo que providencia alguma segurança, mas já não é recomendado o seu uso, a não ser em casos em que haja a necessidade de interoperabilidade com um sistema antigo. Em vez desta cifra, poder-se-ia usar o *SHA-256*, *SHA-3*, *SHA-384* ou *SHA-512* , pois são mais seguras e pertencem à próxima geração de cifras.

Para a troca de chaves e autenticação por HTTPS, em vez de *RSA 2048* e com *SHA-256*, poder-se-ia utilizar as cifras da próxima geração, baseada em curvas elípticas, como o *ECDH-384* para troca de chaves e *ECDSA-384* para autenticação.

6.3 Melhorias para a aplicação móvel

A aplicação realizada está pronta a ser comercializável. Devido à arquitetura da aplicação, é possível juntar novos módulos, que permitam melhorar ainda mais o sistema. As configurações da aplicação permitem alteração para criptografias mais seguras quando o CertVote as suportar.

6.4 Futuro do IVoting

Existem várias implementações para o problema da segurança do voto pela Internet, que permitem tornar as eleições pela Internet cada vez mais seguras. Por outro lado, o número de ataques a sistemas informáticos está no seu auge. Com isto pretende-se dizer que a implementação do voto pela Internet tem um sempre um risco associado.

A grande vantagem do voto pela Internet é a de permitir a realização de eleições de forma mais cómoda, fácil e acessível a todos. Com o aperfeiçoamento dos sistemas de voto, sistemas informáticos, conhecimentos de segurança e criptografia é possível criar sistemas cada vez mais seguros, oferecendo assim a possibilidade de criar eleições cada vez mais seguras. Para além disto, é um método eficiente, e permitindo que o sistema de eleições seja mais utilizado.

Se juntarmos a capacidade de realizar eleições, de forma segura e eficiente, e a capacidade dos eleitores poderem votar com os smartphones, abre-se a possibilidade a que o voto seja cada vez mais utilizado, dando assim mais poder aos eleitores e consequentemente dar-lhes um maior poder de cidadania.

No que diz respeito aos smartphones, a maneira como os sistemas operativos móveis são desenhados, oferecem às aplicações mais mecanismo de segurança do que os sistemas operativos tradicionais tornando-se mais indicadas para realizar o voto.

Como a segurança foi sempre o cerne do qual se desenvolveu este projeto, foi possível ter em conta, ao máximo, os problemas inerentes ao uso de smartphones no voto. Esta é a forma ideal para que este tipo de aplicação seja criada, obtendo assim, uma ferramenta que torna as eleições mais eficientes, não pondo em causa as pedras basilares que lhes subjazem, e.g., veracidade, anonimato do eleitor, entre outros. Na análise das outras aplicações referidas no capítulo 2, com propósito semelhante, verificou-se que os autores das mesmas negligenciaram os mecanismos de segurança dos *smatphones*, propiciando oportunidades de ataque.

Este sistema do voto pela Internet, por exemplo, pode tornar a ideia de democracia em que vivemos hoje, num sistema obsoleto e criar um novo sistema, em que as decisões mais importantes, ou mais específicas, fossem tomadas por todos os eleitores.

Referências

[AaAH11] Hayam K Al-anie, Mohammad A Alia e Adnan A Hnaif. E-VOTING PROTOCOL BASED ON PUBLIC -KEY. *International Journal of Network Security Its Applications (IJNSA) Vol.3 No.4*, 3, 2011. Acedido em : 2015-03-06. URL: http://airccse.org/journal/nsa/0711ijnsa08.pdf.

[ALBD04] Riza Aditya, Byoungcheon Lee, Colin Boyd e Ed Dawson. An efficient mixnet-based voting scheme providing receipt-freeness. In Sokratis Katsikas, Javier Lopez e Günther Pernul, editors, *Trust and Privacy in Digital Business*, volume 3184 of *Lecture Notes in Computer Science*, pages 152–161. Springer Berlin Heidelberg, 2004. URL: http://dx.doi.org/10.1007/978-3-540-30079-3_16.

[Ale14] Alexander N. Pisarchik, Massimiliano Zanin. Chaotic Map Cryptography and Security. *Nova Science Publishers, Inc*, 2014. Acedido em : 2015-03-06. URL: http://www.academia.edu/1912344/Chaotic_Map_Cryptography_and_Security.

[And14] Security Tips | Android Developers. Technical report, 2014. Acedido em : 2015-03-06. URL: http://developer.android.com/training/articles/security-tips.html.

[BDK05] Eli Biham, Orr Dunkelman e Nathan Keller. Related-key boomerang and rectangle attacks. In *Advances in Cryptology–EUROCRYPT 2005*, pages 507–525. Springer, 2005.

[BGP11] Philippe Bulens, Damien Giry e Olivier Pereira. Running mixnet-based elections with helios. In *Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, EVT/WOTE'11, pages 6–6, Berkeley, CA, USA, 2011. USENIX Association. URL: http://dl.acm.org/citation.cfm?id=2028012.2028018.

[Bin08] Timo Bingmann. Speedtest and Comparsion of Open-Source Cryptography Libraries and Compiler Flags - panthema.net. 2008. Acedido em : 2015-03-06. URL: http://panthema.net/2008/0714-cryptography-speedtest-comparison/.

[Bri12] Miguel Brioso. Votação Electrónica Resistente a Vírus Dissertação para a obtenção de Grau de Mestre em Engenharia Informática e Computadores. *Instituto Superior técnico de Lisboa*, 2012. Acedido em : 2015-03-06. URL: https://fenix.tecnico.ulisboa.pt/downloadFile/395144233563/dissertacao.pdf.

REFERÊNCIAS

[Cha84] David Chaum. Blind signature system. In *Advances in cryptology*, pages 153–153. Springer, 1984. Acedido em : 2015-03-06. URL: http://dx.doi.org/10.1007/978-1-4684-4730-9_14.

[Cha01] David Chaum. Surevote: technical overview. In *Proceedings of the workshop on trustworthy elections (WOTE'01)*, 2001.

[Cis14] Cisco. Next Generation Encryption, 2014. Last updated: April 2014 ,Acedido em : 2015-03-06. URL: http://www.cisco.com/web/about/security/intelligence/nextgen_crypto.html.

[DGKV14] Somak Das, Vineet Gopal, Kevin King e Amruth Venkatraman. Iv= 0 security cryptographic misuse of libraries. *Massachusetts Institute of Technology*, 2014. Acedido em : 2015-03-06. URL: https://courses.csail.mit.edu/6.857/2014/files/18-das-gopal-king-venkatraman-IV-equals-zero-security.pdf.

[DMS04] Roger Dingledine, Nick Mathewson e Paul Syverson. Tor: The second-generation onion router. *SSYM'04 Proceedings of the 13th conference on USENIX Security Symposium*, 13, 2004. Acedido em : 2015-03-06. URL: http://portal.acm.org/citation.cfm?id=1251375.1251396.

[EAEZ14] Enas Elbarbary, Ghada Abdelhady, Hussam Elbehiery e Abdelhahim Zekry. Secured and transparent computerized voting system accessible everywhere. *Journal of American Science*, 10(1), 2014. Acedido em : 2015-03-06. URL: http://www.jofamericanscience.org/journals/am-sci/am1001/024_22810am100114_151_157.pdf.

[ElG85] Taher ElGamal. *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, volume 196 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1985. Acedido em : 2015-03-06. URL: http://link.springer.com/chapter/10.1007%2F3-540-39568-7_2#page-1.

[evB14] Phil emon von Bergen. A Mobile Application for Boardroom Voting. *Bern University of applied science, Thesis*, 2014. Acedido em : 2015-03-06. URL: http://e-voting.bfh.ch/app/download/5999765861/vonbergen14.pdf?t=1392211737.

[Gar14] Neha Garg. Comparison of Asymmetric Algorithms in Cryptography. *International Journal of Computer Science and mobile Computing*, 3(4):1190–1196, 2014. Acedido em : 2015-03-06. URL: http://www.ijcsmc.com/docs/papers/April2014/V3I4201499a73.pdf.

[GGI+14] Craig Gentry, Jens Groth, Yuval Ishai, Chris Peikert, Amit Sahai e Adam Smith. Using fully homomorphic hybrid encryption to minimize non-interative zero-knowledge proofs. *Journal of Cryptology*, pages 1–24, 2014. URL: http://dx.doi.org/10.1007/s00145-014-9184-y, doi:10.1007/s00145-014-9184-y.

[Gri15] Pedro Grilo. 2015. Acedido em : 2015-03-06. URL: https://github.com/plusspeed/RSA.

[HSA10] Hatem Hamad, Motaz Saad e Ramzi Abed. Performance evaluation of restful web services for mobile devices. *International Arab Journal of e-Technology*, 1(3):72–78, 2010. Acedido em : 2015-03-06.

[HVM04] Darrel Hankerson, Scott Vanstone e Alfred J Menezes. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2004.

[HWC14] J. Hubbard, K. Weimer e Yu Chen. A study of ssl proxy attacks on android and ios mobile applications. In *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, pages 86–91, Jan 2014. doi:10.1109/CCNC.2014.6866553.

[Inc14] Apple Inc. Security development checklists. 2014. Updated: 2014-02-11 ,Acedido em : 2015-03-06. URL: `https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/SecurityDevelopmentChecklists/SecurityDevelopmentChecklists.html#//apple_ref/doc/uid/TP40002415-CH1-SW1`.

[iSE13] iSECPartners. The ssl conservatory, 2013. Github page , Acedido em : 2015-03-06. URL: `https://github.com/iSECPartners/ssl-conservatory`.

[JZF03] Rui Joaquim, André Zúquete e Paulo Ferreira. Revs–a robust electronic voting system. *IADIS International Journal of WWW/Internet*, 1(2):47–63, 2003.

[KNR10] Dmitry Khovratovich, Ivica Nikolić e Christian Rechberger. Rotational rebound attacks on reduced skein. In *Advances in Cryptology-ASIACRYPT 2010*, pages 1–19. Springer, 2010.

[KT06] Micki Krause e Harold Tipton. *Handbook of information security management*. Taylor & Francis, 2006.

[KZ07] Mirosław Kutyłowski e Filip Zagórski. SCV end to end voting over the Internet. *Institute of Mathematics and Computer Science And Faculty of Fundamental Problems of Technology And Wroclaw University of Technology*, 2007. Accepted to: Lecture Notes in Computer Science – T owards Trustworthy Elections , Acedido em : 2015-03-06. URL: `http://csrc.nist.gov/groups/ST/e2evoting/documents/papers/Zagorski_scv.pdf`.

[LC06] Nuno Santos Luis Costa. Mobile revs - votação eletrónica. *Trabalho final de curso Instituto Superior técnico do curso de licenciatura de Engenharia Informática e de Computadores*, 2006. Acedido em : 2015-03-06. URL: `http://www.gsd.inesc-id.pt/~mrevs/MobileREVS_RelatorioTFC.pdf`.

[LM10] Manfred Lochter e Johannes Merkle. Elliptic curve cryptography (ecc) brainpool standard curves and curve generation, 2010. Acedido em : 2015-03-06. URL: `http://www.hjp.at/doc/rfc/rfc5639.html`.

[MKRS14] M S Sai Mohit, M Karthik, T Rajavel e Ms J Sangeetha. E-Voting System Using Android Application. 2014. International Journal of Research in Engineering and Advanced Technology Volume 2 Issue 2 ISSN: 2320 – 8791 , Acedido em : 2015-03-06. URL: `http://www.ijreat.org/Papers%202014/Issue8/IJREATV2I2060.pdf`.

REFERÊNCIAS

[Mur11] Radovan Murin. Android Powered Portable Voting Device. *Czech Technical University in Prague Faculty of Electrical Engineering Department of Computer Science and Engineering*, 2011. Acedido em : 2015-03-06. URL: https://dip.felk.cvut.cz/browse/pdfcache/murinrad_2011bach.pdf.

[Oos04] Anne-marie Oostveen. Internet Voting Technologies and Civic Participation : The Users ' Perspective. *The public*, XI(1):1–17, 2004. Acedido em : 2015-03-06. URL: http://javnost-thepublic.org/article/pdf/2004/1/4/.

[Pac13] Hewlett Packard. Cyber risk report 2013 . Technical report, 2013. Acedido em : 2015-03-06. URL: http://www8.hp.com/h20195/v2/GetPDF.aspx%2F4AA5-0858ENW.pdf.

[Pei12] Adolfo Peixinho. Secret Sharing Framework based on Digital Certificates. *DOI: 10.13140/2.1.1260.7362 Conference: The 13th European Conference on Cyber Warfare and Security*, 2012. Acedido em : 2015-03-06. URL: http://www.researchgate.net/publication/265057501_Secret_Sharing_Framework_based_on_Digital_Certificates.

[Pro14] Open Web Application Security Project. Projects/OWASP Mobile Security Project - Security Testing Guide - OWASP, 2014. Página web acedida em : 2015-03-06. URL: https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Security_Testing_Guide.

[Rag14] Dhanalakshmi A S Ragunath G, Aarthi R. A + Votz - Google Android Platform for a Mobile - Voting System with Cloud Based Storage and Data hiding Features. *International Journal of Engineering Development and Research (IJEDR), ISSN:2321-9939, Vol.2, Issue 2, pp.2318-2323, June 2014*, 2014. Acedido em : 2015-03-06. URL: http://www.ijedr.org/viewfulltext.php?&p_id=IJEDR1402165.

[ree14] reejosamuel. Simplest rsa wrapper. 2014. Página do github com a framework rsa wrapper , Acedido em : 2015-03-06. URL: https://github.com/reejosamuel/RSA.

[Res00] Certicom Research. Standards for Efficient Cryptography 2 (SEC 2) : Recommended Elliptic Curve Domain Parameters. 2(Sec 2):1–33, 2000. Acedido em : 2015-03-06. URL: http://www.secg.org/SEC2-Ver-1.0.pdf.

[Rit14] Jürg Ritter. Decentralized E-Voting on Android Devices Using Homomorphic Tallying. 2014. Master thesis - Bern University of Applied Sciences Engineering and Information Technology CH-2501 Biel, Switzerland ,Acedido em : 2015-03-06. URL: http://www.cc.gatech.edu/~cpeikert/pubs/FHENIZK.pdf.

[SB12] DouglasW. Jones Simons. e Barbara. *Broken Ballots: Will Your Vote Count?* Stanford University Center for the Study of Language and Information, 2012.

[SEP12] Ryan Sleevi, Chris Evans e Chris Palmer. Public Key Pinning Extension for HTTP, 2012. RFC Internet-Draft , Acedido em : 2015-03-06. URL: http://tools.ietf.org/html/draft-ietf-websec-key-pinning-01.

[SFD+14] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine e J Alex Halderman. Security analysis of the estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014. Acedido em : 2015-03-06. URL: https://jhalderm.com/pub/papers/ivoting-ccs14.pdf.

[Six11] Jeff Six. *Application Security the for Android Plataform*. "O'Reilly Media, Inc.", 2011. A ISBN: 978-1-4493-1507-8 cedido em : 2015-03-06.

[SKW+00] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, Niels Ferguson, Tadayoshi Kohno e Mike Stay. The Twofish Team ' s Final Comments on AES Selection. *AES round*, 2:1–13, 2000. Acedido em : 2015-03-06. URL: https://www.schneier.com/paper-twofish-final.ps.gz.

[SPDS14] Himanshu Shewale, Sameer Patil, Vaibhav Deshmukh e Pragya Singh. ANALYSIS OF ANDROID VULNERABILITIES AND MODERN EXPLOITATION TECHNIQUES. *ICTACT Journal on Communication Technology*, 6948(March):863–867, 2014. Acedido em : 2015-03-06. URL: http://ictactjournals.in/paper/IJCT_Paper_1_863_to_867.pdf.

[VGgTG13] Eliver Pérez Villegas, Gina Gallegos-garcía, Gualberto Aguilar Torres e Héctor Flores Gutiérrez. Implementation of Electronic Voting System in Mobile Phones with Android Operating System. *Journal of Emerging Trends in Computing and Information Sciences*, 4(9):728–737, 2013. Acedido em : 2015-03-06. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.400.194&rep=rep1&type=pdf.

[VK13] VK. 2013. Página do github do código fonte da aplicação móvel do governo da Estonia ,Acedido em : 2015-03-06. URL: https://github.com/vvk-ehk/.

REFERÊNCIAS

Anexo A

Casos de Utilização

O modelo de casos de utilização permite relacionar os atores existentes com as ações que estes podem realizar. É uma representação externa e de alto nível do sistema: representa as funcionalidades do produto acessíveis ao utilizador final.

IDENTIFICADOR	**UC01**
NOME	Usar QR-CODE
DESCRIÇÃO SUMÁRIA	Permite ao visitante utilizar a camera do telemóvel e um código QR-CODE para entrar na página de credenciais
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet.
PÓS-CONDIÇÕES	O visitante entra no ecrã de credenciais
PROCEDIMENTO	1. Carregar no botão "Usar QR-CODE" 2. Apontar câmera para o código

Tabela A.1: Caso de utilização UC01

IDENTIFICADOR	**UC02**
NOME	Introduzir código
DESCRIÇÃO SUMÁRIA	Permite ao visitante introduzir um código único para entrar na eleição
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet.
PÓS-CONDIÇÕES	O visitante entra no ecrã de credenciais
PROCEDIMENTO	1. Preencher campo de texto. 2. Carregar no botão "Submeter código".

Tabela A.2: Caso de utilização UC02

IDENTIFICADOR	**UC03**
NOME	Introduzir credenciais
DESCRIÇÃO SUMÁRIA	Permite ao eleitor aceder ao seu boletim de voto
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet e escolhido uma votação aberta.
PÓS-CONDIÇÕES	O eleitor entra no ecrã no ecrã de preencher boletins
PROCEDIMENTO	1. Preencher campos de texto. 2. Carregar no botão "Submeter credenciais".

Tabela A.3: Caso de utilização UC03

IDENTIFICADOR	**UC04**
NOME	Preencher boletim
DESCRIÇÃO SUMÁRIA	Permite escolher a sua opção de voto
ATOR	Eleitor
PRÉ-CONDIÇÕES	Ter acesso à Internet e estar autenticado.
PÓS-CONDIÇÕES	O eleitor escolhe um candidato
PROCEDIMENTO	1. Carregar com o dedo na *checkbox* do candidato em que pretende votar.

Tabela A.4: Caso de utilização UC04

IDENTIFICADOR	**UC05**
NOME	Navegar entre boletim
DESCRIÇÃO SUMÁRIA	Permite navegar entre boletins
ATOR	Eleitor
PRÉ-CONDIÇÕES	Ter acesso à Internet e estar autenticado.
PÓS-CONDIÇÕES	Muda de boletim
PROCEDIMENTO	1. Fazer *swipe* para a esquerda ou para a direita.

Tabela A.5: Caso de utilização UC05

IDENTIFICADOR	**UC06**
NOME	Ver opções de voto
DESCRIÇÃO SUMÁRIA	Permite ver o voto
ATOR	Eleitor
PRÉ-CONDIÇÕES	Ter acesso à Internet e estar autenticado.
PÓS-CONDIÇÕES	Ecrã de submeter voto
PROCEDIMENTO	1. Fazer *swipe* para a direita até ao ecrã de submeter voto.

Tabela A.6: Caso de utilização UC06

IDENTIFICADOR	**UC07**
NOME	Submeter voto
DESCRIÇÃO SUMÁRIA	Permite submeter o voto
ATOR	Visitante
PRÉ-CONDIÇÕES	Ter acesso à Internet e estar autenticado.
PÓS-CONDIÇÕES	Apresentação se o voto foi bem sucedido ou não e redirecionamento para a página inicial
PROCEDIMENTO	1. Carregar no botão submeter voto.

Tabela A.7: Caso de utilização UC07

Anexo B

Resposta JSON dos serviços

```
1
2 ******** RESPOSTA *********
3 {"mensagem":{
4   "mensagem":"A eleicao aberta",
5   "status":"OK"
6 },
7 "electionResponse":{
8   "staus":"OK",
9   "template":"election-ready",
10  "promotorKey":"-----BEGIN PUBLIC KEY-----MIIBIjANB ...-----END PUBLIC KEY-----","
        image":"
        iVBORw0KGgoAAAANSUhEUgAAAGkAAAAxCAYAAADZX4egAAAABmJLR0QA ... ",
11  "electionName":"SIBS",
12  "electionTitle":"SIBS",
13  "electionDescription":"Sociedade Interbancaria de Servicos",
14  "credentialNames":["EMAIL"],
15  "electionCode":"SIBS",
16  "election":true}
17 }
```

Código B.1: Resposta JSON S01

```
1
2 ******** PEDIDO   *********
3
4  {   "credentials":"[{\"name\":\"EMAIL\",\"value\":\"pgrilo\"}]",
5   "electionCode":"SIBS",
6   "password":"ola123",
7   "publicKey":"-----BEGIN PUBLIC KEY-----
        MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDQYqxgCSmrrw7QzydzwpO+Khm7xpCYhlWPnz0+\
        n1eSR64YRDth9hWQ69tlSZDSu5bvXvV8EhfmuW+OUGI1m9Wnmakn1/Ox77Ct69/uo3CeiLFlu+yEX
```

```
      \npIKjB1e57qPD2q0l1QqVVZlI2XMMhVfNN6uhAy3QRkdI+35TUOl6u0wkTQIDAQAB\n-----END
      PUBLIC KEY-----",
  "signature":"
      bddfc52b7d6bd9b07c86555e31f2be2e08f5ec7b907f8d9516b639b2d97a95d56a8c42b89b84c44069bea3c75023ddbe6
      "
  }


******** RESPOSTA *********
{"mensagem":
  {"mensagem":"Login com sucesso",
  "status":"OK"},
 "vr":
  {"boletins":[
    {"bulletinId":1,
    "description":"Boletim Geral",
    "options":[
      {"description":"Lista A",
      "idVoteOption":1,
      "image":" . . .",
      "voteOptionCount":0},
      {"description":"Lista B",
      "idVoteOption":2,
      "image":" ...u003d\u003d","
      voteOptionCount":0}]
    }
```

Código B.2: Resposta JSON S02

```
{"mensagem":{
"mensagem":"A eleicao aberta",
"status":"OK"
},
"electionResponse":{
"staus":"OK",
"template":"election-ready",
"promotorKey":"-----BEGIN PUBLIC KEY-----MIIBIjANB ...-----END PUBLIC KEY-----","
    image":"
    iVBORw0KGgoAAAANSUhEUgAAAGkAAAAxCAYAAADZX4egAAAABmJLR0QA ... ",
"electionName":"SIBS",
"electionTitle":"SIBS",
"electionDescription":"Sociedade Interbancaria de Servicos",
"credentialNames":["EMAIL"],
"electionCode":"SIBS",
"election":true}
}
```

Código B.3: Resposta JSON S03